Московский государственный технический университет
Кафедра «Системы автоматического управления»

Методы вычислений

Андрей Леонидович Масленников


Вычисленные методы.
Амосов А.А., Дубинский Ю.А. Копченова Н.В.
И.: Лань, 2014, 672 с.


Численные методы.
Андреев В.Б.
И.: Макс-пресс, 2013, 337 с.


Численные методы.
Бахвалов Н.С., Жидков Н.П., Кобельков Г.М.
И.: Бином. Лаборатория знаний, 2017, 640 с.


Численные методы линейной алгебры. Лабораторный практикум.
Белов С.А., Злотых Н.Ю.
И.: Нижегородский государственный университет, 2005, 58 с.


Основы численных методов.
Вержбицкий В.М.
И.: Высшая школа, 2002, 840 с.

Матрицы и вычисления.
Воеводин В.В., Кузнецов А.Ю.
И.: Наука, 1984, 320 с.

Оптимизация.
Галеев Э.М., Тихомиров В.М.
И.: Эдиториал УРСС, 2000, 320 с.

Вычислительная линейная алгебра. Теория и приложения.
Деммель Дж.
И.: Мир, 2001, 430с.

Математические основы теории автоматического управления. В трех томах.
Иванов В.А., Медведев В.С., Чемоданов Б.К., Ющенко А.С.
3-е издание. И.: Высшая школа, 2008.

Численные методы алгебры.
Семушкин М.В.
И.: Ульяновский государственный технический университет, 2006, 176.


Вычислительные методы линейной алгебры.
Фаддеев Д.К., Фаддеева В.Н.
4-е издание. И.: Лань, 2009, 736 с.

Вычислительная задача

Вычислительная задача - это одна из трех типов математических задач решение которой необходимо получить численно.

Принципиальные типы математических задач:

- прямая;
  (непосредственное вычисление решения)
- обратная;
  (Вычисление параметров математической модели)
- задача идентификации.
  (идентификация математической модели)

Модель

Идеализированное представление, обладающее определенной (достаточной для решения задачи моделирования) степению адекватности (подобия), исследуемой системы (объекта, процесса).

Символьные (аналитические) методы

Аналитические методы решения математических задач - это методы получения решения в буквенно-символьном виде включая разлиные математические преобразования исходной задачи.

Аналитические подходы решения задач, как правило:

- не эффективны по быстродействию получения решения;
- имеют более низкую точность полученного решения;
- не могут быть применены в автономных технических системах;
- не применимы для решения современных сложных технических задач.

Численные методы

Численные методы - это алгоритмы и их реализации для решения математических задач, когда получаемый результат получается в виде, как правило, набора чисел.

С развитием вычислительной техники, как правило, под получаемым набором чисел понимают набор цифровых данных, хранящихся в том или ином виде в памяти вычислителя.

Численные методы - это не только альтернатива символьным (аналитическим) подходам при решении математических задач, но и в подавляющем большинстве случаев - единственный реализуемый подход решения этих задач.

Аналитические подходы решения задач, как правило:

- не эффективны по быстродействию получения решения;
- имеют более низкую точность полученного решения;
- не могут быть применены в автономных технических системах.

Требования, предъявляемые к численным методам:

- по требованиям к разработке алгоритмов:
  - достаточный уровень быстродействия;
    (численное решение задачи должно завершаться за ограниченное время, отведенное на это решение обобщенной задачей)
  - достижимость (устойчивость/сходимость) решения;
    (численное решение задачи должно быть близким к ожидаемому с теоретической точкой зрения и желательно быть единственным)
  - минимальность ошибки (погрешности) вычислений;
    (вычислительная ошибка, с которой получается достижимое итоговое решение должна быть сведена к минимуму)
- по оптимальности реализации алгоритма:
  - минимальность временной вычислительной сложности;
    (время, затрачиваемое вычисления должно сводиться к минимуму)
  - минимальность пространственной вычислительной сложности.
    (использование вычислительных ресурсов - объема используемой ОЗУ или процессорного времени должно сводится к минимуму)

Классификация численных методов:

- методы эквивалентных преобразований;
- методы аппроксимации;
- прямые методы;
- итерационные методы;
- стохастические методы.

Характеристики вычислительных задач и методов:

- погрешности получаемого решения;
- корректность и обусловленность задачи;
- устойчивость и/или сходимость решения.

Виды численных методов по решаемым задачам:

- методы векторно-матричных преобразований и разложений;
- методы решения линейных и нелинейных систем уравнений;
- методы интерполяции функций;
- методы интегрирования функций;
- методы дифференцирования функций;
- методы решения задач оптимизации;
  (непосредственный численный поиск экстремумов)
- методы решения задачи Коши;
  (методы интегрирования систем дифференциальных уравнений)

Методы эквивалентных преобразований

В этих методах исходная задача (ее постановка, начальные данные, математическое описание) заменяется на другую, имеющую тоже решение. Как правило, метод используется тогда, когда в исходной постановке задача не может быть решена, либо в новой постановке процесс получения решения более эффективен.

В данном случае речь не идет об уменьшении погрешности вычислений, хотя численно, де-факто, это возможно.

В большей степени метод свойственен для аналитических подходов, но для численных явным примером может служит замена исходной задачи вычисления обратной матрицы на решение этой же задачи но с использованием матричных разложений.

Методы аппроксимации

В этих методах исходная задача (ее постановка, начальные данные, матетическое описание) заменяется (аппроксимируется) на другую, имеющую решение близкое к исходной. Неточность аппроксимации проявляется в появлении в результатах вычислений ошибки аппроксимации.

Если ошибка аппроксимации стремится к нулю при стремлении параметров метода аппроксимации к некоторому значению, то говорят, что аппроксимация сходится.

Примером метода аппроксимации является вычисление определенного интеграла, когда подынтегральная функция заменяется на последовательность более простых функций. Аналогично происходит и при решении задач интерполяции и решении задачи Коши.

Отдельным классом методов аппроксимации можно считать задачи линеаризации нелинейных функционалов.

Прямые методы

В этих методах решение задачи получается за конечное число вычислительных операций, которое в общем случае зависит от размера входных и данных.

Большинство прямых методов было разработано до эпохи компьютеризации, и не смотря на то, что их алгоритм достаточно легко программируется, как правило, эти методы при решении практических задач практически не используются, т.к. очень чувствительны к ошибкам округления.

Например, решение СЛАУ методом Крамера для любой матрицы 3×3 потребует одинакового числа операций.

Итерационные методы

В этих методах решение задачи получается с помощью построения итерационного процесса, т.е. некоторого итерационного процесса приближения к истинному решению, где количество итераций, в теоретическом смысле, не фиксировано.

Итерационный процесс, в зависимости от решаемой задачи может завершаться при выполнении одного или нескольких условий, например:

- количество итераций превысило заданное максимальное значение;
- разность решений полученных на текущей и предыдущей итерациях меньше некоторого заданного значения;
- разность некоторых функционалов полученных на текущей и предыдущей итерациях меньше некоторого заданного значения.

Для итерационных методов важно свойство сходимости метода к истинному решению. Сходимость зависит от многих факторов, но, как правило, связывается со значениями начальных условий. Множество начальных условий называется областью сходимости, если для них метод сходится.

Стохастические методы

В этих методах решение задачи получается с помощью моделирования случайных экспериментов, как правило, многократного, и построении (вычислении) статистических оценок.

Теоретически стохастические методы могут выдавать решение с большой точностью, но при этом затраченное время становится слишком большим. Однако для радя задач, особенно при наличии случайных составляющих в постановке задачи, стохастические методы являются единственно применимыми.

Примером стохастических методов могут служить метода имитационного моделирования Монте-Карло для вычисления интегралов сложных фигур. Эти методы, по сути, представляют собой концепцию многократного повторения одного и того же процесса, но с различной реализацией стохастических процессов.

Погрешность определения решения

Погрешность (в скалярном или векторном виде) определения решения математической задачи численным методом - это величина отклонения полученным решением y и истинным y_0.

Погрешности принципиально бывают трех видов:

- абсолютная;
- относительная;
- приведенная.

Абсолютная погрешность

Абсолютная погрешность Δy - это погрешность следующего вида:

$$\Delta y = |y - y_0|$$

где:

- y - полученное решение;
- y_0 - истинное решение, в общем случае неизвестно.

Использование абсолютной погрешности далеко не всегда удобно, т.к. ее значение зависит от принятых единиц измерения и масштабов величин.

Относительная погрешность

Относительная погрешность δy - это погрешность следующего вида, при условии, что полученное решение не нулевое:

$$\delta y = \frac{|y - y_0|}{y} = \frac{\Delta y}{y}$$

где:

- y - полученное решение;
- y_0 - истинное решение, в общем случае неизвестно.

Значение относительной погрешности не зависит от принятых единиц измерения и масштабов величин и может быть выражено в процентах.

Верхние и нижние границы погрешностей

На практике вычисление Δy и δy невозможно, т.к. неизвестно y. Поэтому, задача определения погрешностей сводится к их оценке по косвенным признакам и получении верхних границ погрешностей, т.е.:

$$|y - y_0| \leqslant \bar{\Delta} y$$
$$\frac{|y - y_0|}{y} \leqslant \bar{\delta} y$$

Понятие нижних границ погрешностей в численных методах не вводится, но используется несколько в более сложном контексте в задачах теории оценивания стохастических величин.

Верхние и нижние границы погрешностей, часто называют предельными погрешностями.

Приведенная погрешность

Приведенная погрешность - это отношение величины верхней границы абсолютиной погрешности к некотором нормирующему значению:

$$\gamma = \frac{\bar{\Delta} y}{y_n},$$

где y_n никак не связано с вычислениями, а задается как некоторая константа.

Наиболее часто в качестве нормируюшего значения выступает количество точек данных, по которым считается величина погрешности. Например, это очень часто реализуется в процессе решения оптимизационных задач или вычислении дискретного преобразования Фурье.

Точность решения

Точность решения в качественных рассуждениях - это противоположность погрешности, однако на практике, как правило, речь идет об одних и тех же характеристиках.

Например, «получить решение СЛАУ с точностью ε», что означает, что погрешность вычислений не должна превышать значения ε.

Разделение на абсолютную и относительную точность не проводится, полагая, что из контекста задачи ясно, какой вид вычисления погрешности используется. На практике, как правило, принято считать в относительных погрешностях.

Значащие цифры

Значащие цифры числа y - это все (и целые и дробные) цифры в его записи, начиная с первой не нулевой для числа, заданного в виде конечной десятичной дроби:

$$y = \alpha_n\, \alpha_{n-1} \cdots \alpha_0 \,.\, \beta_1\, \beta_1 \cdots \beta_m$$

Верные значащие цифры

Верные значащие цифры числа y - это те цифры разряд которых меньше величины верхней границы абсолюиной погрешности, т.е., например:

$$\Delta y = 2 \times 10^{-6}$$
$$y = 0.010300$$

означает, что число y имеет четыре верные значащие цифры.

Количество верных значащих цифр связано с величиной относительной погрешности:

- если число y содержит N верных значащих цифр, то справедливо неравенство:
$$\delta y \leqslant (10^{N-1} - 1)^{-1} \approx 10^{1-N}$$
- для того, чтобы число y содержало N верных значащих цифр, необходимо, чтобы было выполнено неравенство:
$$\delta y \leqslant (10^{N} + 1)^{-1} \approx 10^{-N}$$
- если число y имеет ровно N верных значащих цифр, то справедливо неравенство:
$$10^{-N-1} \leqslant \delta y \leqslant 10^{N+1}, \quad \text{т.е.} \quad \delta y \approx 10^{-N}$$

Например, пусть $y = 3.14$, а $y_0 = 3.14159$, тогда:

$$\Delta y = |y - y_0| = 0.00159$$
$$\bar{\Delta} y = 0.0016$$
$$\bar{\delta} y = 0.0016/3.14 = 0.0051$$
$$\delta y = 10^{-3} = 0.001$$

Пусть решается задача определения величины y по вычисленному x:

$$x = 0.999997$$
$$N_x = 6$$
$$\delta x = 10^{-6}$$
$$y = 1 - x = 1 - 0.999997 = 0.000003$$
$$N_y = 1$$
$$\delta y = 10^{-1}$$

В результате получаем, что при достаточно высокой точности определения величины x точность определения величины y оказалось очень маленькой.

$$y = 2 - x = 2 - 0.999997 = 1.000003$$
$$N_y = 7$$
$$\delta y = 10^{-7}$$

Это пример одновременно и влияния ошибок округления и влияния на точность вычислений количества разрядов в числе с плавающей точкой

Абсолютная погрешность функции нескольких переменных

Абсолютная погрешность Δy при

$$y = f(\mathbf{x})$$

где $f(\mathbf{x})$ дифференцируемая в некоторой области G функция m переменных, вычисление которой производится по приближенно полученным значениям $\mathbf{x}$, определяется следующим образом:

$$\Delta y = \sum_{j=1}^{m} \max_{[\mathbf{x}_0, \mathbf{x}]} \left| \frac{\partial f(\mathbf{x})}{\partial x_j} \right| \Delta x$$

Верхняя граница абсолютной ошибки примет вид:

$$\bar{\Delta} y = \sum_{j=1}^{m} \left| \frac{\partial f(\mathbf{x})}{\partial x_j} \right| \bar{\Delta} x$$

Причины возникновения погрешностей при численном решении задач:

- неточность используемой математической модели;
  (например в случае аппроксимации или линеаризации)
- неточность задания начальных значений;
  (в меньшей степени относится непосредственно к численному методу)
- неточность, заложенная в самом методе;
  (например, в итерационных методах)
- погрешности связанные с машинной арифметикой;
  (в памяти вычислителя есть конечная разрешающая способность при хранении чисел с плавающей точкой)
- ошибки округления.
  (в большей степени относятся к вводу и выводу данных)

Погрешность метода

Погрешность метода $\delta_{\text{м}} y$ - это погрешность, источником которой является сам метод.

Например, в численных методах вычисления определенных интегралов подынтегральная функция аппроксимируется последовательностью более простых функций, но они не в точности повторяют характер исходной подынтегральной кривой.

Погрешность округления

Погрешность округления или ошибка округления $\delta_{\text{о}} y$ - это погрешность обусловленная ограничением на ввод и вывод знаков дробной части чисел с плавающей точкой. Эта погрешность не связана непосредственно с вычислениями, а связана только с вводом и выводом данных.

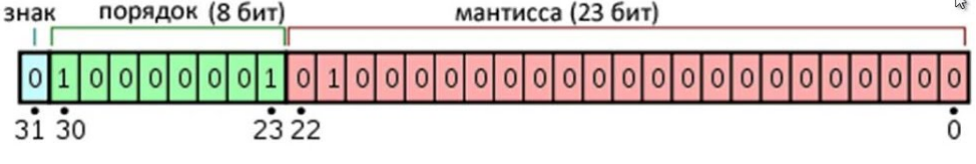
Погрешность связанная с машинной арифметикой

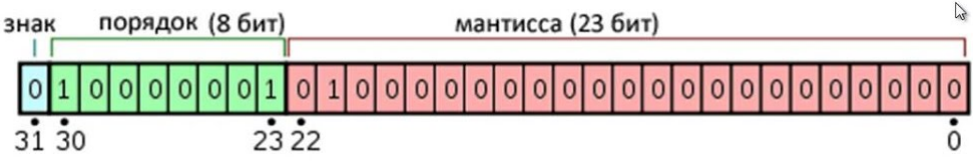
Погрешность вычислений связанной с машинной арифметикой $\delta_{\text{в}}y$ - это погрешность обусловленная разрешающей способностью вычислителя при хранении и выполнении операций над числами с плавающей точкой.

Существует стандарт IEEE 754, который описывает арифметику чисел с плавающей точкой в вычислителях различных типов, например:

- число одинарной точности;
  (32 бита / 4 байта на число)
- число двойной точности;
  (64 бита / 8 байт на число)

В памяти числа, как правило, хранятся согласно данной схеме:





Формула определения числа:

$$y = (-1)^S \cdot C \cdot b^Q$$

где b - основание (2 или 10), а S равно 1 для отрицательных чисел, а порядок Q смещен, т.е. старший бит порядка отвечает за знак, а C - мантисса.

Как правило, представление числа с плавающей точек реализуется с помощью нормализованной мантиссы, т.е. само число всегда начинается с единицы, т.е. 1.xxxx, в этом случае количество верных значащих бит:

$$N_{y,32b} = 24$$
$$N_{y,64b} = 53$$
$$N_{y,256b} = 237$$

Для чисел одинарной точности с нормализованной мантисой получаем:

$$\delta y \approx 2^{-N_{y,32b}} = 2^{-24} \approx 5.96 \times 10^{-8}$$
$$N_y = 7$$

Для чисел двойной точности с нормализованной мантисой получаем:

$$\delta y \approx 2^{-N_{y,64b}} = 2^{-53} \approx 1.11 \times 10^{-16}$$
$$N_y = 15$$

Таким образом, например при вычислении π при одинарной точности:

$$\pi = 3.141592\underbrace{653589793238462643}_{\text{мусор}}$$

Таким образом, например при вычислении π при двойной точности:

$$\pi = 3.14159265358979\underbrace{3238462643}_{\text{мусор}}$$

Неустранимые погрешности

Неустранимыми погрешности δy - это погрешности, которые не зависят от параметров численного метода, т.е. по сути, это неточность используемой математической модели и неточность задания начальных условий. Такие погрешности еще называют ошибками моделирования.

По большому счету, в разряд неустранимых нужно также внести и погрешности, связанные с машинной арифметикой, однако, на текущий момент развития вычислительной техники эти погрешности для большинства практических задач малы и ими можно пренебречь.

Устранимые погрешности

Устранимые погрешности - это погрешности, которые можно устранить или снизить с помощью настройки численного метода решения.

Например, погрешность вычисления определенного интеграла можно снизить повысив количество разбиений рассматриваемого интервала.

Корректность задачи

Корректность задачи (сформулировано Адамаром и Петровским) - это характеристика математической задачи, для которой выполняются все следующие свойства:

- решение задачи существует при любых входных данных;
  (например, поиск обратной матрицы невозможен для исходной матрицы с отрицательным детерминантом)
- решение обладает свойством единственности;
  (множество решений не представляет никакого интереса для последующего анализа, при этом в большинстве практических задач, особенно оптимизационных, единственность решения недостижима)
- решение устойчиво по отношению к малым изменениям входных данных.
  (под устойчивостью понимается, что решение не уходит в $\pm\infty$, а локализовано в некоторой δ-окрестности)

Некорректность задачи, как правило, означает неверно выбранную математическую модель, неверно заданные или некорректные начальные условия и неверно выбран численный метод решения и/или его параметры.

Абсолютная устойчивость решения

Решение y вычислительной задачи называется аболютно устойчивым по входным данным x, если , существует $\varepsilon > 0$ и $\delta = \delta(\varepsilon) > 0$, что исходным данным x, лежащим в δ-окрестности ($\Delta x < \delta$), соответствует приближенное решение y, лежащее в ε-окрестности ($\Delta y < \varepsilon$).

Δx и Δy - абсолютные погрешности, а δ и ε можно трактовать как точность. Соответственно, желая повысить точность решения, необходимо более точно задавать начальные данные (сужать окрестности).

Неустойчивость решения означает, что для сколь угодно малой δ-окрестности, можно найти такие начальные данные x, что решение y выйдет за пределы ε-окрестности. На практике, это, как правило, обозначает, что решение в процессе вычислений постепенно стремится к $\pm\infty$.

Относительная устойчивость решения

Решение y вычислительной задачи называется относительно устойчивым по входным данным x, если , существует $\varepsilon > 0$ и $\delta = \delta(\varepsilon) > 0$, что исходным данным x, лежащим в δ-окрестности ($\delta x < \delta$), соответствует приближенное решение y, лежащее в ε-окрестности ($\delta y < \varepsilon$).

δx и δy - относительные погрешности, а δ и ε можно трактовать как точность. Соответственно, желая повысить точность решения, необходимо более точно задавать начальные данные (сужать окрестности).

Обусловленность вычислительной задачи

Обусловленность задачи - это характеристика математической задачи, которая показывает насколько решение задачи чувствительно по отношению к погрешностям входных данных.

Вычислительная задача называется хорошо обусловленной, если малым погрешностям входных данных соответствуют малые погрешности решения и плохо обусловленной, если малым погрешностям входных данных могут соответствовать сильные изменения в решении.

Число обусловленности

Число обусловленности - это коэффициент, характеризующий возможное возрастание погрешностей решения по отношению к погрешностям входных данных, т.е.:

$$\Delta y = \nu_{\Delta} \Delta x$$
$$\delta y = \nu_{\delta} \delta x$$

где ν_{Δ} - абсолютное число обусловленности, а ν_{δ} - относительное число обусловленности, которые в том числе могут быть выражены и через предельные погрешности.

Для плохо обусловленных задач $\nu \gg 1$. Например, при $\nu_{\delta} \approx 10^N$ порядок N показывает число верных цифр в решении, которое может быть утеряно по отношению к числу верных цифр в начальных данных.

Вектор:

$$\mathbf{v} = \begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_n \end{bmatrix}$$

Матрица:

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1m} \\ a_{21} & a_{22} & \cdots & a_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nm} \end{bmatrix}$$

Квадратная матрица

Квадратная матрица - это матрица, количество строк и столбцов которой одинаково и равно n:

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{bmatrix}$$

Верхняя и нижняя треугольные матрицы

Верхней треугольной или нижней треугольной называются матрицы следующего вида (диагональные элементы - ненулевые):

$$\mathbf{L} = \begin{bmatrix} a_{11} & 0 & \cdots & 0 \\ a_{21} & a_{22} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{bmatrix}, \quad \mathbf{U} = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ 0 & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & a_{nn} \end{bmatrix}$$

Обратная матрица

Обратной матрицей $\mathbf{A}^{-1}$ для квадратной матрицы $\mathbf{A}$ является матрица, произведение которой на исходную равно единичной матрице:

$$\mathbf{A}\mathbf{A}^{-1} = \mathbf{A}^{-1}\mathbf{A} = \mathbf{I}$$

Невырожденная матрица

Невырожденная (неособенная, обратимая) матрица - это матрица, для которой существует обратная матрица. Невырожденность обеспечивается следующими условиями:

- строки и столбцы матрицы линейно независимы;
- ранг матрицы равен ее размерности;
- определитель матрицы не равен нулю.

Матрица, не удовлетворяющая этим условиям, называется вырожденной (сингулярной, необратимой).

Транспонированная матрица

Для матрицы $\mathbf{A}$ размерностью $n \times m$ транспонированной матрицей $\mathbf{A}^\top$ будет матрица с размерностью $m \times n$, где каждый элемент матрицы $\mathbf{A}^\top$ определяется следующим образом:

$$\mathbf{A}^\top_{ij} = \mathbf{A}_{ji}$$

Симметричная матрица

Квадратная матрица $\mathbf{A}$ является симметричной, если:

$$\mathbf{A} = \mathbf{A}^\top$$

Кососимметричная матрица

Квадратная матрица $\mathbf{A}$ является кососимметричной (при условии, что диагональные элементы равны нулю), если:

$$\mathbf{A} = -\mathbf{A}^\top$$

- двойное транспонирование:

$$(\mathbf{A}^\top)^\top = \mathbf{A}$$

- транспонирование суммы:

$$(\mathbf{A} + \mathbf{B})^\top = \mathbf{A}^\top + \mathbf{B}^\top$$

- транспонирование произведения:

$$(\mathbf{AB})^\top = \mathbf{B}^\top \mathbf{A}^\top$$

- транспонирование матрицы умноженной на скаляр:

$$(\mathbf{A}\,\lambda)^\top = \lambda\,\mathbf{A}^\top$$

- транспонирование обратной матрицы:

$$(\mathbf{A}^{-1})^\top = (\mathbf{A}^\top)^{-1}$$

- определитель транспонированной матрицы:

$$\det(\mathbf{A}) = \det(\mathbf{A}^\top)$$

Эрмитова матрица

Эрмитова матрица - это квадратная матрица, в общем случае состоящая из комплексных чисел, которая равна своей эрмитово-сопряженной (сопряженно-транспонированной матрице) и равна транспонированной комплексно-сопряженной матрице, т.е.:

$$\mathbf{A} = (\bar{\mathbf{A}})^\top = \mathbf{A}^*,$$

где $\bar{z} = a - \jmath b$ - комплесно-сопряженное число для $z = a + \jmath b$, применяемый к каждому элементу матрицы.

Пример:

$$\mathbf{A} = \begin{bmatrix} 5 & 2+\jmath \\ 2-\jmath & 7 \end{bmatrix} \quad \bar{\mathbf{A}} = \begin{bmatrix} 5 & 2-\jmath \\ 2+\jmath & 7 \end{bmatrix} \quad (\bar{\mathbf{A}})^\top = \begin{bmatrix} 5 & 2+\jmath \\ 2-\jmath & 7 \end{bmatrix} = \mathbf{A}^*$$

$\mathbf{A}^*$ это и транспонирование и поэлементное комплексное сопряжение

Унитарная матрица

Унитарная матрица - это квадратная матрица, в общем случае состоящая из комплексных чисел, произведение которой на эрмитово-сопряженную матрицу равно единичной матрице, т.е.:

$$\mathbf{U}^* \mathbf{U} = \mathbf{U} \mathbf{U}^* = \mathbf{I}$$

Ортогональная матрица

Ортогональная матрица - это унитарная матрица, но только с вещественными числами, для которой справедливо:

$$\mathbf{A}^\top \mathbf{A} = \mathbf{A} \mathbf{A}^\top = \mathbf{I},$$

и как следствие:

$$\mathbf{A}^\top = \mathbf{A}^{-1}$$

Нормальная матрица

Нормальная матрица - это квадратная матрица $\mathbf{A}$, в общем случае состоящая из комплексных числел, для которой справедливо следующее соотношение:

$$\mathbf{A}^* \mathbf{A} = \mathbf{A} \mathbf{A}^*,$$

где $\mathbf{A}^*$ - сопряженно-транспонированная матрица, а для матрицы с исключительно вещественными числами:

$$\mathbf{A}^\top \mathbf{A} = \mathbf{A} \mathbf{A}^\top$$

Основные свойства нормальной матрицы:

- нормальная матрица диагонализируема через унитарные матрицы, т.е. представима в виде:

  $$\mathbf{A} = \mathbf{U} \mathbf{D} \mathbf{V}$$

  где $\mathbf{D}$ - диагональная матрица.
- нормальная матрица с действительными собственными значениями является эрмитовой матрицей.

Разреженная матрица

Разреженная матрица - это матрица, которая преимущественно состоит из нулевых значений, что зачастую проявляется в задачах решения систем дифференциальных уравнений в частных производных. Такие системы называеются плохообусловленными или некорректно поставленными.

“Разреженность” существенно усложняет процесс поиска решения (может приводить к конфликту больших и малых чисел и/или вырожденности матрицы), а также требует большого объема оперативной памяти для своего хранения.

В качестве решения проблемы использования разреженных матриц является применение метода регуляризации Тихонова, который используется в задачах поиска численного решения СЛАУ и оптимизационных задачах.

Матричная экспонента

Матричная экспонента (экспоненциальная матрица) $\mathrm{e}^{\mathbf{A}}$ - это матричная функция от квадратной матрицы, которая определяется степенным рядом следующим образом:

$$\mathrm{e}^{\mathbf{A}} = \sum_{k=0}^{\infty} \frac{1}{k!} \mathbf{A}^k$$

Матричная экспонента обладает рядом свойств:

- $\mathrm{e}^{\mathbf{0}} = \mathbf{I}$
- $\mathrm{e}^{\mathbf{A}^\top} = (\mathrm{e}^{\mathbf{A}})^\top$
- $\mathrm{e}^{\mathbf{Y}\mathbf{X}\mathbf{Y}^{-1}} = \mathbf{Y}\,\mathrm{e}^{\mathbf{X}}\,\mathbf{Y}^{-1}$, если $\mathbf{Y}$ - невырожденная

Квадратный корень из матрицы

Матрица $\mathbf{B}$ называется квадратным корнем из матрицы $\mathbf{A}$, если:

$$\mathbf{B}\,\mathbf{B} = \mathbf{A}$$

Методы вычисления квадратного корня из матрицы $\mathbf{A}$:

- через явные формулы;
  (только для малых порядков)
- с использованием спектрального разложения матрицы, а затем:

$$\sqrt{\mathbf{A}} = \mathbf{V}\,\sqrt{\mathbf{D}}\,\mathbf{V}^{-1}$$

  (проблемы нахождения спектрального разложения)
- с использованием разложения в форму Жордана;
- метод Денмана-Биверса;
  (сходимость не гарантирована)
- вавилонский метод.
  (сходимость не гарантирована)
- и др.

Метод Денмана-Биверса

Итерационный метод вычисления квадратного корня (в том числе и для матрицы), т.е. матрицы $\mathbf{B}$ для матрицы $\mathbf{A}$, для которого после инициализации:

$$\mathbf{Y}_0 = \mathbf{A}$$
$$\mathbf{Z}_0 = \mathbf{I}$$

запускается итерационный процесс (с фиксированным, т.е. заданным максимальным количеством итераций), на каждой итерации которого вычисляются:

$$\mathbf{Y}_{k+1} = 0.5\left(\mathbf{Y}_k + \mathbf{Z}_k^{-1}\right)$$
$$\mathbf{Z}_{k+1} = 0.5\left(\mathbf{Z}_k + \mathbf{Y}_k^{-1}\right)$$
$$\mathbf{B}_{k+1} = 2\,\mathbf{B}_{k-1} - \mathbf{B}_k\,\mathbf{Z}_k\,\mathbf{B}_k$$

где $\mathbf{B}_0 = \mathbf{Z}_{k-1}^{-1}$.

Вавилонский метод

Итерационный метод вычисления квадратного корня (в том числе и для матрицы), т.е. матрицы $\mathbf{B}$ для матрицы $\mathbf{A}$, для которого после инициализации:

$$\mathbf{B}_0 = \mathbf{I}$$

запускается итерационный процесс (с фиксированным, т.е. заданным максимальным количеством итераций), на каждой итерации которого вычисляется:

$$\mathbf{B}_{k+1} = 0.5 \left(\mathbf{B}_k + \mathbf{A}\,\mathbf{B}_k^{-1} \right)$$

Преимуществом вавилонского метода является вычисление только одной обратной матрицы на каждом шаге, однако его устойчивость и сходимость существенно хуже, чем у метода Денмана-Биверса.

Минор

Минор M порядка k матрицы $\mathbf{A}$ - это определитель квадратной матрицы, составленной из элементов матрицы $\mathbf{A}$ с номерами строк и столбцов от 1 до k включительно, например для $k = 3$:

$$M = \det \begin{bmatrix} 1 & 2 & 3 \\ 4 & 1 & 6 \\ 7 & 8 & 1 \end{bmatrix} = 104$$

Главный минор

Главный минор M матрицы $\mathbf{A}$ - это минор матрицы $\mathbf{A}$ для которого совпадают выбранные номера строк и столбцов.

Угловой минор

Угловой минор M матрицы $\mathbf{A}$ порядка k - это минор, состоящий из первых k строк и k столбцов матрицы $\mathbf{A}$.

Базисный минор

Базисным минором матрицы $\mathbf{A}$ называется не равный нулю минор максимального порядка (т.е. все миноры более высокого порядка равным нулю). Важным следствием является то, что все строки (столбцы), формирующие этот минор, являются линейно независимыми.

Ранг матрицы

Ранг матрицы $r = \text{rang}(\mathbf{A})$ - это скалярное значение, соответствующее порядку базисного минора матрицы $\mathbf{A}$. Если ранг матрицы больше нуля, то в ней r линейно независимых столбцов и строк.

Линейная независимость

Система векторов $\left\{ \mathbf{v}_1, \ldots, \mathbf{v}_n \right\}$ является линейно независимой, т.е.ни один вектор не может быть выражен через другой, в том случае, если уравнение:

$$\alpha_1 \mathbf{v}_1 + \ldots + \alpha_n \mathbf{v}_n = 0$$

имеет тривиальное решение только при

$$\alpha_1 = \ldots = \alpha_n = 0.$$

Дополнительный минор

Дополнительный минор $\bar{M}_{ij}$ квадратной матрицы **A** - это определитель, составленный из элементов матрицы **A** путем удаления строк и столбцов, например, при удалении второй строки и третьего столбца:

$$\bar{M}_{23} = \det \begin{bmatrix} 1 & 2 & \boxdot \\ \boxdot & \boxdot & \boxdot \\ 7 & 8 & \boxdot \end{bmatrix} = \det \begin{bmatrix} 1 & 2 \\ 7 & 8 \end{bmatrix} = 8 - 4 = 4$$

Алгебраическое дополнение

Алгебраическое дополнение A_{ij} элемента a_{ij} матрицы **A** это скалярное значение, определяемое следующим образом:

$$A_{ij} = (-1)^{i+j} \bar{M}_{ij}$$

Присоединенная матрица

Присоединенная матрица $\operatorname{adj}\mathbf{C}$ - это квадратная матрица, составленная из алгебраических дополнений матрицы $\mathbf{A}^\top$, т.е.:

$$\operatorname{adj}\mathbf{C} = \begin{bmatrix} A_{11} & A_{21} & \cdots & A_{n1} \\ A_{12} & A_{22} & \cdots & A_{n2} \\ \vdots & \vdots & \ddots & \vdots \\ A_{1n} & A_{2n} & \cdots & A_{nn} \end{bmatrix}$$

Используя присоединенную матрицу можно вычислить обратную матрицу:

$$\mathbf{A}^{-1} = \frac{1}{\det \mathbf{A}} \operatorname{adj}\mathbf{C},$$

однако, этот способ вычислительно не эффективен и на практике заменяется другими методами.

Собственные вектора и собственные значения матрицы

Собственным вектором $\mathbf{x}$ и соответствующим ему собственном значением λ матрицы $\mathbf{A}$ называются величины, для которых следующее равенство имеет ненулевое решение:

$$\mathbf{A}\,\mathbf{v} = \lambda\,\mathbf{v}$$

Сигнулярные значения матрицы

Сингулярные значения σ матрицы $\mathbf{A}$ - это вещественные неотрицательные числа, которые удовлетворяют следующему равенству при векторах $\mathbf{v}$ и $\mathbf{u}$ единичной длины:

$$\mathbf{A}\,\mathbf{v} = \sigma\,\mathbf{u}$$

В общем случае определение собственных значений и собственных векторов требует решения матричного уравнения:

$$(\mathbf{A} - \lambda \mathbf{I})\,\mathbf{v} = 0\,,$$

что аналитически решается только для малых матриц, а численные алгоритмы не решают задачу целиком, а находят либо несколько собственных значений, либо применимы только для определенного типа матриц.

Можно использовать спектральное разложение:

$$\mathbf{A} = \mathbf{V}\,\mathbf{\Lambda}\,\mathbf{U}^*\,,$$

но этот подход не всегда применим. Альтернативой могут быть другие виды разложения матриц.

Положительно определенная матрица

Положительно определенной матрицей $\mathbf{A} \succ 0$ называется эрмитова матрица, которая удовлетворяет любому из следующих условий (приведены не все возможные):

- для всех ненулевых комплексных векторов:
$$\mathbf{z}^* \mathbf{A} \mathbf{z} > 0$$
- все собственные значения матрицы $\mathbf{A}$ положительны, т.е.
$$\lambda_i(\mathbf{A}) > 0$$
- определители всех угловых миноров положительны.
(критерий Сильвестра)

Положительная определенность очень важна для решения оптимизационных задач с матричной функцией потерь

Положительно полуопределенная матрица

Положительно полуопределенной матрицей $\mathbf{A} \succcurlyeq 0$ называется эрмитова матрица, которая удовлетворяет аналогичным условиям, но со знаком “$\geqslant$” в критериях.

Отрицательно определенная матрица

Отрицательно определенной матрицей $\mathbf{A} \prec 0$ называется эрмитова матрица, которая удовлетворяет аналогичным условиям, но со знаком “$<$” в критериях.

Отрицательно полуопределенная матрица

Отрицательно полуопределенной матрицей $\mathbf{A} \preccurlyeq 0$ называется эрмитова матрица, которая удовлетворяет аналогичным условиям, но со знаком “$\leqslant$” в критериях.

Норма матрицы

Норма матрицы - это скалярное значение, характеризующее меру объема матрицы (насколько в ней большие значения), по аналогии с нормой вектора, которая показывает длину этого вектора.

Операторная норма

Максимальная сумма значений в столбцах:

$$\|\mathbf{A}\|_1 = \max_{1 \leqslant j \leqslant m} \sum_{i=1}^{n} |a_{ij}|$$

Максимальная сумма значений в строках:

$$\|\mathbf{A}\|_\infty = \max_{1 \leqslant i \leqslant n} \sum_{j=1}^{m} |a_{ij}|$$

Спектральная норма $\|\mathbf{A}\|_2$ связана с $\max \sigma(\mathbf{A})$.

Векторная p-норма

В общем виде определяется следующим образом:

$$\|\mathbf{A}\|_p = \left(\sum_{i=1}^{n} \sum_{j=1}^{m} |a_{ij}|^p \right)^{1/p}$$

Норма Фробениуса (евклидова норма)

Это частный случай p-нормы с $p = 2$, т.е.:

$$\|\mathbf{A}\|_F = \sqrt{\sum_{i=1}^{n} \sum_{j=1}^{m} a_{ij}^2}$$

при этом

$$\|\mathbf{A}\|_F^2 = \sum_{i=1}^{n} \sum_{j=1}^{m} a_{ij}^2 = \operatorname{tr}(\mathbf{A}\,\mathbf{A}^*) = \sum_{k} \sigma_k^2(\mathbf{A})$$

След матрицы

След матрицы $\mathrm{tr}(\mathbf{A})$ - это скалярное значение (по факту, операция), которое равно сумме всех диагональных элементов этой матрицы, т.е.

$$\mathrm{tr}(\mathbf{A}) = \sum_{i=1}^{\infty} a_{ii}$$

След матрицы обладает рядом свойств:

- след матрицы - это линейная операция:

$$\mathrm{tr}(\alpha\,\mathbf{A} + \beta\,\mathbf{B}) = \alpha\ \mathrm{tr}(\mathbf{A}) + \beta\ \mathrm{tr}(\mathbf{B})$$

- след матрицы равен следу транспонированной матрицы:

$$\mathrm{tr}(\mathbf{A}) = \mathrm{tr}\left(\mathbf{A}^{\top}\right)$$

- след матрицы равен сумме ее собственных значений:

$$\mathrm{tr}(\mathbf{A}) = \sum_{i} \lambda_i$$

`eig` - собственные значения и собственные вектора матрицы;
`eigs` - собственные значения и собственные вектора матрицы;
`svds` - сингулярные значения и сингулярные вектора матрицы;
`expm` - экспоненциальная матрица;
`sqrtm` - квадратный корень из матрицы;
`inv` - вычисление обратной матрицы (не рекомендуется);
`diag` - диагональная матрица (или диагональные элементы матрицы);
`eye` - единичная матрица;
`gallery` - генерация особых (специфических) матриц;
`tril` - нижняя треугольная часть матрицы;
`triu` - верхняя треугольная часть матрицы;
`norm` - норма матрицы или вектора;
`sum` - суммирование элементов вектора;
`nnz` - количество ненулевых элементов в матрице или векторе.

Матричные разложения

Разложение (декомпозиция) матрицы - это получение представления исходной матрицы в виде других матриц. Использование полученных разложений может существенно сократить объемы матричных вычислений в практических задачах.

Основные виды матричных разложений:

- на базе собственных значений и векторов;
  - спектральное разложение;
  - разложение в Жорданову нормальную форму;
  - разложение Шура (с вариациями);
  - сингулярное разложение.
- получаемые специальными алгоритмами.
  (QR-, LL-, LU-разложения и др.)

Спектральное разложение матрицы

Спектральное разложение матрицы $\mathbf{A}$ - это ее представление в виде произведения трех матриц:

$$\mathbf{A} = \mathbf{V}\,\mathbf{\Lambda}\,\mathbf{V}^{-1},$$

где:

- $\mathbf{V}$ - матрица с собственными векторами матрицы $\mathbf{A}$;
- $\mathbf{\Lambda}$ - диагональная матрица с собственными значениями $\lambda(\mathbf{A})$.

Только матрицы, имеющие полный набор собственных значений, могут быть представлены в виде спектрального разложения.

Существует несколько подходов вычисления спектрального разложения матрицы, которые требуют больших вычислительных ресурсов.

Каноническая форма Жордана

Каноническая форма Жордана - это представление квадратной матрицы $\mathbf{A}$ в виде:

$$\mathbf{A} = \mathbf{C}\,\mathbf{J}\,\mathbf{C}^{-1},$$

где:

- $\mathbf{C}$ - матрица перехода к новому базису;
- $\mathbf{J}$ - матрица Жордана.

По сути, это разложение - обобщение спектрального разложения на случай кратных собственных значений.

Если кратность всех собственных значений матрицы $\mathbf{A}$ равна единице, то матрица $\mathbf{J}$ - диагональная. В противном случае матрица $\mathbf{J}$ является блок-диагональной и состоит из Жордановых блоков.

Каноническая форма Жордана $\mathbf{J}$ может иметь следующий вид:

$$\mathbf{J} = \begin{bmatrix} \lambda_1 & 1 & 0 & 0 & 0 & 0 \\ 0 & \lambda_1 & 1 & 0 & 0 & 0 \\ 0 & 0 & \lambda_1 & 0 & 0 & 0 \\ 0 & 0 & 0 & \lambda_2 & 0 & 0 \\ 0 & 0 & 0 & 0 & \lambda_3 & 1 \\ 0 & 0 & 0 & 0 & 0 & \lambda_3 \end{bmatrix} = \begin{bmatrix} \mathbf{J}_1 & 0 & 0 \\ 0 & \mathbf{J}_2 & 0 \\ 0 & 0 & \mathbf{J}_3 \end{bmatrix},$$

где соответствующие Жордановы блоки определены следующим образом:

$$\mathbf{J}_1 = \begin{bmatrix} \lambda_1 & 1 & 0 \\ 0 & \lambda_1 & 1 \\ 0 & 0 & \lambda_1 \end{bmatrix}, \quad \mathbf{J}_2 = \begin{bmatrix} \lambda_2 \end{bmatrix}, \quad \mathbf{J}_3 = \begin{bmatrix} \lambda_3 & 1 \\ 0 & \lambda_3 \end{bmatrix}$$

Разложение Шура

Разложение Шура матрицы $\mathbf{A}$ - это ее представление в виде произведения трех матриц:

$$\mathbf{A} = \mathbf{U}\mathbf{T}\mathbf{U}^*,$$

где:

- $\mathbf{U}$ - унитарная матрица;
- $\mathbf{U}^*$ - эрмитово-сопряженная матрица;
- $\mathbf{T}$ - верхняя треугольная матрица с $t_{ii} = \lambda_i(\mathbf{A})$.

Для случая вещественных чисел разложение имеет вид:

$$\mathbf{A} = \mathbf{V}\mathbf{T}\mathbf{V}^\top,$$

где:

- $\mathbf{V}$ - ортогональная матрица.

Сингулярное разложение

Сигнулярное разложение (SVD) матрицы $\mathbf{A}$ - это ее представление в виде произведения трех матриц:

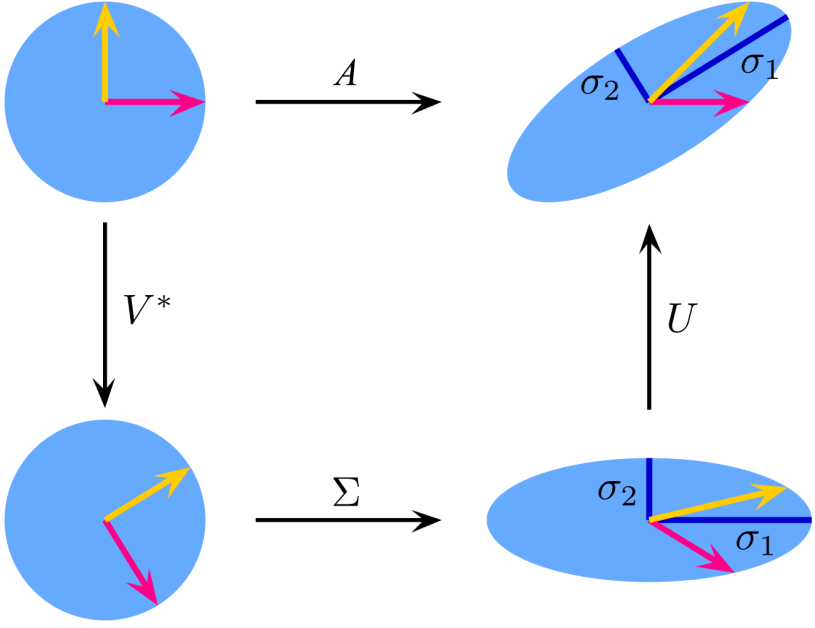
$$\mathbf{A} = \mathbf{U}\,\Sigma\,\mathbf{V}^*,$$

где:

- $\mathbf{U}$ - унитарная матрица;
- $\mathbf{V}$ - унитарная матрица;
- Σ - диагональная матрица с сингулярными значениями $\mathbf{A}$.

$\mathbf{U}$ состоит из левых сингулярных векторов матрицы $\mathbf{A}\mathbf{A}^*$
$\mathbf{V}$ состоит из правых сингулярных векторов матрицы $\mathbf{A}^*\mathbf{A}$

SVD может использоваться для нахождения псевдо обратных матриц
Алгоритмы вычисления сингулярного разложения сложны


A
σ2
σ1
V*
U
Σ
σ2
σ1

LU-разложение

LU-разложение матрицы **A** - это ее представление в виде произведения двух матриц:

$$\mathbf{A} = \mathbf{L}\,\mathbf{U},$$

Разложение существует при неравенстве нулю всех главных миноров матрицы **A**.

Методы вычисления *LU*-разложения:

- метод Гаусса;
  (низкая точность для разреженных матриц, большой объем вычислений)
- алгоритм Дулиттла;
  *(**L** - унитреугольная, **U** - верхняя треугольная)*
- алгоритм Кроута.
  *(**L** - нижняя треугольная, **U** - унитреугольная)*

Алгоритм Дулиттла:

В результате применения алгоритма **L** получается нижнкй унитреугольной, а **U** - верхней треугольной. Для $i = 1 \ldots n$ вычисляем элементы матриц **U** и **L**:

$$u_{ij} = a_{ij} - \sum_{k=1}^{i-1} l_{ik} u_{kj}, \quad j = i, \ldots, n$$

$$l_{ii} = 1$$

$$l_{ji} = \frac{1}{u_{ii}} \left(a_{ji} - \sum_{k=1}^{i-1} l_{jk} u_{ki} \right), \quad j = i+1, \ldots, n$$

Method employed in the solution of normal equations and the adjustment of a triangularization.
Doolittle, M.H.
U.S. Coast and Geodetic Survey, Report. pp. 115–120. 1878

Алгоритм Кроута:

В результате применения алгоритма **L** получается нижней треугольной, а **U** - верхней унитреугольной. Для $i = 1 \ldots n$ вычисляем элементы матриц **U** и **L**:

$$l_{ji} = a_{ji} - \sum_{k=1}^{i-1} l_{jk} u_{ki}, \quad j = i, \ldots, n$$

$$u_{ii} = 1$$

$$u_{ij} = \frac{1}{l_{ii}} \left(a_{ij} - \sum_{k=1}^{i-1} l_{ik} u_{kj} \right), \quad j = i+1, \ldots, n$$

Вычисление определителя:

$$\det(\mathbf{A}) = \det(\mathbf{LU}) = \det(\mathbf{L})\,\det(\mathbf{U}) = \left(\prod_{i=1}^{n} l_{ii}\right)\left(\prod_{i=1}^{n} u_{ii}\right)$$

Для решения СЛАУ:

$$\mathbf{A}\,\mathbf{x} = \mathbf{b}$$

$$\mathbf{L}\,\mathbf{U}\,\mathbf{x} = \mathbf{b}$$

последовательно решаем два уравнения (методом подстановки):

$$\mathbf{L}\,\mathbf{y} = \mathbf{b}$$

$$\mathbf{U}\,\mathbf{x} = \mathbf{y}$$

Считая **x** матрицей **X** и **b** единичной матрицей **I** можно определить обратную матрицу $\mathbf{A}^{-1}$, но это не самый эффективный способ

LUP-разложение

LUP-разложение это *LU*-разложения матрицы **A** с частичным выбором ведущего элемента (перестановка по строкам), которое записывается следующим образом:

$$\mathbf{PA} = \mathbf{LU},$$

а, в случае перестановок по столбцам *LUP*-разложение записывается в виде:

$$\mathbf{AQ} = \mathbf{LU},$$

а, в наиболее общем виде:

$$\mathbf{PAQ} = \mathbf{LU},$$

где:

- **P** - матрица перестановок по строкам;
- **Q** - матрица перестановок по столбцам;
- **L** - нижняя унитреугольная матрица;
- **U** - верхняя треугольная матрица.

LDU-разложение

LDU-разложение матрицы **A** - это ее представление в виде произведения трех матриц:

$$\mathbf{A} = \mathbf{L}\,\mathbf{D}\,\mathbf{U},$$

где:

- **D** - диагональная матрица;
- **L** - нижняя унитреугольная матрица;
- **U** - верхняя унитреугольная матрица.

LL-разложение

LL-разложение положительно определенной матрицы $\mathbf{A}$ - это ее представление в виде произведений двух матриц:

$$\mathbf{A} = \mathbf{L}\mathbf{L}^*$$
$$\mathbf{A} = \mathbf{U}^*\mathbf{U}$$

где:

- $\mathbf{L}$ - нижняя треугольная матрица;
- $\mathbf{U}$ - верхняя треугольная матрица.

Для вещественных матриц: $\mathbf{L}^* = \mathbf{L}^\top$ и $\mathbf{U}^* = \mathbf{U}^\top$.

Положительная определенность матрицы $\mathbf{A}$ является необходимым условием для получения разложения, а симметричность матрицы $\mathbf{A}$ позволяет получить единственно верное разложение, при котором результат $\mathbf{L}\mathbf{L}^*$ будет равен исходной матрице $\mathbf{A}$.

Существует несколько алгоритмов получения разложения Холецкого:

- классический алгоритм Холецкого;
  (в его основе метод исключения Гаусса при приведении исходной матрицы к нижнетреугольному виду)
- алгоритм Холецкого-Банахевича;
  (алгоритм вычисления матрицы L начиная с первой строки и далее итерационно по каждой строке)
- алгоритм Холецкого-Кроута;
  (алгоритм вычисления матрицы U начиная с первого столбца и далее итерационно по каждому столбцу)

Алгоритм Кроута в обобщеном виде является алгоритмом вычисления *LU*-разложения, где **L** - нижняя треугольная, а **U** - унитарная верхняя треугольная.

Формулы для нахождения **L** в комплексном случае:

$$l_{ij} = \frac{1}{l_{jj}} \left(a_{ij} - \sum_{k=1}^{j-1} l_{ik} l_{jk}^{*} \right)$$

$$l_{ii} = \sqrt{a_{ii} - \sum_{k=1}^{i-1} l_{ik}\, l_{ik}^{*}}$$

Формулы для нахождения **L** в вещественном случае:

$$l_{ij} = \frac{1}{l_{jj}} \left(a_{ij} - \sum_{k=1}^{j-1} l_{ik} l_{jk} \right)$$

$$l_{ii} = \sqrt{a_{ii} - \sum_{k=1}^{i-1} l_{ik}^{2}}$$

Исходные данные:

$$\mathbf{A} = \begin{bmatrix} 1.4 & 1 & 1 \\ 1 & 0.9 & 1 \\ 1 & 1 & 1.4 \end{bmatrix}$$

Полученный результат:

$$\mathbf{L} = \begin{bmatrix} 1.18 & 0 & 0 \\ 0.85 & 0.43 & 0 \\ 0.85 & 0.66 & 0.5 \end{bmatrix}$$

Процесс решения:

- $i = 1,\ j = 0 \Rightarrow \ldots$ - считать нечего;
- $l_{11} = \sqrt{1.4 - 0} = 1.18$
- $l_{21} = \dfrac{1}{1.18}(1 - 0) = 0.85$
- $l_{22} = \sqrt{0.9 - 0.85^2} = 0.43$
- $l_{31} = \dfrac{1}{1.18}(1 - 0) = 0.85$
- $l_{32} = \dfrac{1}{0.43}(1 - 0.85^2) = 0.66$
- $l_{33} = \sqrt{1.4 - (0.85^2 + 0.66^2)} = 0.5$

Для несимметричной матрицы **A**:

$$\mathbf{A} = \begin{bmatrix} 1.4 & 1 & 2 \\ 1 & 0.9 & 1 \\ 1 & 1 & 1.4 \end{bmatrix}$$

Матрица **L** получится в виде:

$$\mathbf{L} = \begin{bmatrix} 1.18 & 0 & 0 \\ 0.85 & 0.43 & 0 \\ 0.85 & 0.66 & 0.5 \end{bmatrix}$$

А результат разложения

$$\mathbf{L}\mathbf{L}^\top = \begin{bmatrix} 1.4 & 1 & 1 \\ 1 & 0.9 & 1 \\ 1 & 1 & 1.4 \end{bmatrix} \neq \mathbf{A}$$

Для несимметричных матрицы разложение Холецкого принципиально можно получить, но оно не будет соответствовать исходной матрице **A**

LDL-разложение

LDL-разложение (модификация *LL*-разложения) положительно определенной матрицы **A** - это ее представление в виде произведения трех матриц:

$$\mathbf{A} = \mathbf{L}\,\mathbf{D}\,\mathbf{L}^{*}$$

где:

- **D** - диагональная матрица;
- **L** - нижняя унитреугольная матрица.

При вычислении *LDL*-разложения, в отличии от *LL*-разложения, не требуется вычисление квадратных корней, а условием существования разложения является отличие от нуля всех угловых миноров исходной матрицы **A**.

Формулы для нахождения **L** и **D** в комплексном случае:

$$l_{ij} = \frac{a_{ij} - \sum\limits_{k=1}^{j-1} l_{ik}\, d_k\, l_{jk}^*}{d_{jj}}$$

$$d_{jj} = a_{jj} - \sum_{k=1}^{j-1} l_{kj}\, l_{kj}^*\, d_{kk}$$

Формулы для нахождения **L** и **D** в вещественном случае:

$$l_{ij} = \frac{a_{ij} - \sum\limits_{k=1}^{j-1} l_{ik}\, d_k\, l_{jk}}{d_{jj}}$$

$$d_{jj} = a_{jj} - \sum_{k=1}^{j-1} l_{jk}^2\, d_{kk}$$

Исходные данные:

$$\mathbf{A} = \begin{bmatrix} 1.4 & 1 & 1 \\ 1 & 0.9 & 1 \\ 1 & 1 & 1.4 \end{bmatrix}$$

Полученный результат:

$$\mathbf{L} = \begin{bmatrix} 1.0 & 0 & 0 \\ 0.7143 & 1.0 & 0 \\ 0.7143 & 1.5385 & 1.0 \end{bmatrix}$$

$$\mathbf{D} = \begin{bmatrix} 1.4 & 0 & 0 \\ 0 & 0.1857 & 0 \\ 0 & 0 & 0.2462 \end{bmatrix}$$

Процесс решения:

- $l_{11} = 1.0$
- $d_{11} = 1.4 - 0 = 1.4$
- $l_{22} = 1.0$
- $l_{21} = \dfrac{1.0 - 0}{1.4} = 0.7143$
- $d_{22} = 0.9 - 0.7143 = 0.1857$
- $l_{33} = 1.0$
- $l_{31} = \dfrac{1.0 - 0}{1.4} = 0.7143$
- $l_{32} = \dfrac{1.0 - 0.7143}{0.1857} = 1.5385$
- $d_{33} = 1.4 - 0.7143 - 0.4396 = 0.2462$

Разложение Холецкого очень часто используется при решении следующих задач:

- при решении систем линейных алгебраических уравнений, где подход аналогичен рассмотренному ранее с *LU*-разложением;
  (точное решение гарантировано только при симметричной **A***)*
- в нелинейных методах оптимизации для получения положительно определенных матриц в процессе поиска решений решения;
- в методах моделирования Монте-Карло для генерации случайных величин коррелируемых, между собой;
  (в методах, где необходимо получать ковариационную матрицу, которая должна быть положительно определенной и симметричной)
- как более эффективный метод вычисления обратной матрицы для не эрмитовой матрицы, при чем результат - обратная матрица, получится эрмитовой;


Principes d'une nouvelle technique de la méthode des moindres carrés; Méthode de résolution numérique des équations linéaires, du calcul des déterminants et des inverses et de réduction des formes quadratiques.
Banachiewicz, T.
Bull. Inter. Acad. Polon. Sci., Sér. A, 393–404 p. 1938


The life and work of André Cholesky.
Brezinski, C.
Numerical Algorithms, vol. 43, no. 3, pp 279-288. 2006

QR-разложение

QR-разложение матрицы **A** (не обязательно квадратной) - это ее представление в виде произведения двух матриц:

$$\mathbf{A} = \mathbf{Q}\,\mathbf{R}$$

где:

- **Q** - унитарная матрица;
- **R** - верхняя треугольная матрица.

Единственное разложение можно получить, только если наложить ограничение положительности диагональных элементов матрицы **R**.

Классический алгоритм Грама-Шмидта

Матрица $\mathbf{A}$ рассматривается в виде:

$$\mathbf{A} = \left[\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_n\right]$$

Оператор проекции определяется следующим образом:

$$\text{proj}_{\mathbf{b}}\,\mathbf{a} = \frac{\mathbf{a}\cdot\mathbf{b}}{\mathbf{b}\cdot\mathbf{b}}\,\mathbf{b}$$

Алгоритм начинается с последовательного вычисления векторов $\mathbf{b}_k$:

$$\begin{aligned}
\mathbf{b}_1 &= \mathbf{a}_1 \\
\mathbf{b}_2 &= \mathbf{a}_2 - \text{proj}_{\mathbf{b}_1}\,\mathbf{a}_2 \\
\mathbf{b}_3 &= \mathbf{a}_3 - \text{proj}_{\mathbf{b}_1}\,\mathbf{a}_3 - \text{proj}_{\mathbf{b}_2}\,\mathbf{a}_3 \\
&\ldots \\
\mathbf{b}_k &= \mathbf{a}_k - \sum_{j=1}^{k-1} \text{proj}_{\mathbf{b}_j}\,\mathbf{a}_k
\end{aligned}$$

Затем вычисляется матрица $\mathbf{Q}$, каждый столбец которой представляет собой нормированный вектор $\mathbf{b}_k$:

$$\mathbf{Q} = \left[\frac{\mathbf{b}_1}{\|\mathbf{b}_1\|}, \frac{\mathbf{b}_2}{\|\mathbf{b}_2\|}, \ldots, \frac{\mathbf{b}_n}{\|\mathbf{b}_n\|} \right]$$

После чего вычисляется матрица $\mathbf{R}$:

$$\mathbf{R} = \mathbf{Q}^\top \mathbf{A}$$

Ввиду наличия ошибок округления при вычислении векторов $\mathbf{b}_j$ эти вектора становятся не точно ортогональными, в следствии чего, процедура ортогонализации теряет свой смысл и применение классического алгоритма Грама-Шмидта приводит к существенным ошибкам, а он сам считается вычисленно неустойчивым. В качестве альтернативы можно использовать:

- модифицированный алгоритм Грама-Щмидта;
- метод Хаусхолдера (метода разложений);
- метод Гивенса (метод вращений).

Для заданной матрицы $\mathbf{A}$:

$$\mathbf{A} = \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}$$

Последовательно вычисляем вектора $\mathbf{b}_k$:

$$\mathbf{b}_1 = \mathbf{a}_1 = \begin{bmatrix} 2 \\ 1 \end{bmatrix}$$

$$\mathbf{b}_2 = \mathbf{a}_2 - \text{proj}_{\mathbf{b}_1} \mathbf{a}_2$$

$$= \begin{bmatrix} 1 \\ 2 \end{bmatrix} - \frac{2 \cdot 1 + 1 \cdot 2}{2 \cdot 2 + 1 \cdot 1} \begin{bmatrix} 2 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 \\ 2 \end{bmatrix} - \frac{4}{5} \begin{bmatrix} 2 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 \\ 2 \end{bmatrix} - \begin{bmatrix} 1.6 \\ 0.8 \end{bmatrix} = \begin{bmatrix} -0.6 \\ 1.2 \end{bmatrix}$$

Вычисляем нормы векторов $\mathbf{b}_k$:

$$\|\mathbf{b}_1\| = 2.2361, \quad \|\mathbf{b}_2\| = 1.3416$$

С использованием полученных векторов $\mathbf{b}_k$:

$$\mathbf{b}_1 = \begin{bmatrix} 2 \\ 1 \end{bmatrix}, \quad \mathbf{b}_2 = \begin{bmatrix} -0.6 \\ 1.2 \end{bmatrix}$$

И их норм:

$$\|\mathbf{b}_1\| = 2.2361, \quad \|\mathbf{b}_2\| = 1.3416$$

Вычисляем матрицу $\mathbf{Q}$:

$$\mathbf{Q} = \begin{bmatrix} 0.8944 & -0.4472 \\ 0.4472 & 0.8944 \end{bmatrix}$$

Вычисляем матрицу $\mathbf{R}$:

$$\mathbf{R} = \mathbf{Q}^\top \mathbf{A} = \begin{bmatrix} 2.2361 & 1.7889 \\ 0.0000 & 1.3416 \end{bmatrix}$$

В модифицированном алгоритме Грама-Шмидта повышается вычислительная устойчивость за счет изменения способа вычисления векторов $\mathbf{b}_j$.

Исходный вариант:

$$\mathbf{b}_1 = \mathbf{a}_1$$
$$\mathbf{b}_2 = \mathbf{a}_2 - \operatorname{proj}_{\mathbf{b}_1} \mathbf{a}_2$$
$$\mathbf{b}_3 = \mathbf{a}_3 - \operatorname{proj}_{\mathbf{b}_1} \mathbf{a}_3 - \operatorname{proj}_{\mathbf{b}_2} \mathbf{a}_3$$
$$\dots$$
$$\mathbf{b}_k = \mathbf{a}_k - \sum_{j=1}^{k-1} \operatorname{proj}_{\mathbf{b}_j} \mathbf{a}_k$$

Модифицированный вариант:

$$\mathbf{b}_k^{(1)} = \mathbf{a}_k - \operatorname{proj}_{\mathbf{b}_1} \mathbf{a}_k$$
$$\mathbf{b}_k^{(2)} = \mathbf{b}_k^{(1)} - \operatorname{proj}_{\mathbf{b}_2} \mathbf{b}_k^{(1)}$$
$$\dots$$
$$\mathbf{b}_k^{(k-1)} = \mathbf{b}_k^{(k-2)} - \operatorname{proj}_{\mathbf{b}_{k-1}} \mathbf{b}_k^{(k-2)}$$
$$\mathbf{b}_k = \frac{\mathbf{b}_k^{(k-1)}}{\left\| \mathbf{b}_k^{(k-1)} \right\|}$$

Каждый вектор $\mathbf{b}_k^{(j)}$ ортогонален вектору $\mathbf{b}_k^{(j-1)}$ независимо от ошибок округления. Набор векторов $\mathbf{b}_k$ сразу формируют матрицу $\mathbf{Q}$.

Метод Хаусхолдера

В основе метода Хаусхолдера (метод отражений) лежит преобразование Хаусхолдера - это такое линейное преобразование матрицы **A** (векторного пространства, образуемого столбцами матрицы), которое описывает отражение (векторного пространства) относительно некоторой гиперплоскости, проходящей через начало координат.

Преобразование может использоваться также для получения преобразования исходной матрицы в трехдиагональную.

Метод Хаусхолдера более вычислительно устойчив по сравнению с ортогонализацией Грама-Шмидта, но менее вычислительно эффективен чем метод Гивенса.

Инициализация:

$$\mathbf{Q} = \mathbf{I}$$
$$\mathbf{R} = \mathbf{A}$$

На каждой итерации для $k = 1, \ldots, n-1$:

$$\begin{aligned}
\mathbf{x} &= \mathbf{a}_k \\
x_{1,\ldots,k-1} &= 0 \\
\mathbf{u} &= \mathbf{x} - \|\mathbf{x}\| \cdot \mathbf{e} \\
\mathbf{P} &= \mathbf{I} - \frac{2\,\mathbf{u}\,\mathbf{u}^\top}{\|\mathbf{u}\|^2} \\
\mathbf{Q} &= \mathbf{Q}\,\mathbf{P} \\
\mathbf{A} &= \mathbf{P}\,\mathbf{A} \\
\mathbf{R} &= \mathbf{A}
\end{aligned}$$

где $\mathbf{e}$ - орт, у которого на каждой итерации только k-ый элемент равен единице.

На $k-1$-ой итерации в $\mathbf{A}$ всегда получаются нули ниже главной диагонали во всех столбцах до $k-1$ включительно, т.е. преобразования необходимо продолжать только над главным минором $k, \ldots, n$, что можно реализовать через обнуление элементов вектора $\mathbf{x}$:

Для исходной матрицы:

$$\mathbf{A} = \begin{bmatrix} 1.4 & 1 & 1 \\ 1 & 0.9 & 1 \\ 1 & 1 & 1.4 \end{bmatrix}$$

для $k = 1$ и вычислим:

$$\mathbf{x} = \begin{bmatrix} 1.4 \\ 1 \\ 1 \end{bmatrix}$$

$$\|\mathbf{x}\| = 1.99$$

$$\mathbf{u} = \begin{bmatrix} 1.4 \\ 1 \\ 1 \end{bmatrix} - 1.99 \cdot \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} -0.59 \\ 1 \\ 1 \end{bmatrix}$$

для $k = 1$ получим значения матриц:

$$\mathbf{P} = \mathbf{I} - \frac{2\,\mathbf{u}\,\mathbf{u}^{\top}}{\|\mathbf{u}\|^2} = \begin{bmatrix} 0.7035 & 0.5025 & 0.5025 \\ 0.5025 & 0.1482 & -0.8518 \\ 0.5025 & -0.8518 & 0.1482 \end{bmatrix}$$

$$\mathbf{Q} = \mathbf{Q}\mathbf{P} = \mathbf{I}\mathbf{P} = \mathbf{P} = \begin{bmatrix} 0.7035 & 0.5025 & 0.5025 \\ 0.5025 & 0.1482 & -0.8518 \\ 0.5025 & -0.8518 & 0.1482 \end{bmatrix}$$

$$\mathbf{A} = \mathbf{P}\mathbf{A} = \begin{bmatrix} 1.9900 & 1.6583 & 1.9096 \\ 0 & -0.2158 & -0.5417 \\ 0 & -0.1158 & -0.1417 \end{bmatrix}$$

$$\mathbf{R} = \mathbf{A}$$

Как видно, первый столбец $\mathbf{A}$ уже приведен к треугольному виду, следовательно нужно преобразовывать только главный минор из 2 и 3 строчек и столбцов.

для $k = 2$ и вычислим:

$$\mathbf{x} = \begin{bmatrix} 1.6583 \\ -0.2158 \\ -0.1158 \end{bmatrix}$$

$$x_1 = 0$$

т.е.:

$$\mathbf{x} = \begin{bmatrix} 0 \\ -0.2158 \\ -0.1158 \end{bmatrix}$$

$$\|\mathbf{x}\| = 0.2449$$

$$\mathbf{u} = \begin{bmatrix} 0 \\ -0.2158 \\ -0.1158 \end{bmatrix} - 0.2449 \cdot \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ -0.4608 \\ -0.1158 \end{bmatrix}$$

для $k = 2$ получим значения матриц:

$$\mathbf{P} = \mathbf{I} - \frac{2\,\mathbf{u}\,\mathbf{u}^{\top}}{\|\mathbf{u}\|^2} = \begin{bmatrix} 1 & 0 & 0 & \\ 0 & -0.8811 & -0.4729 & \\ 0 & -0.4729 & & 0.8811 \end{bmatrix}$$

$$\mathbf{Q} = \mathbf{Q}\,\mathbf{P} = \begin{bmatrix} 0.7035 & -0.6804 & 0.2052 \\ 0.5025 & 0.2722 & -0.8206 \\ 0.5025 & 0.6804 & 0.5334 \end{bmatrix}$$

$$\mathbf{A} = \mathbf{P}\,\mathbf{A} = \begin{bmatrix} 1.9900 & 1.6583 & 1.9096 \\ 0 & 0.2449 & 0.5443 \\ 0 & 0 & 0.1313 \end{bmatrix}$$

$$\mathbf{R} = \mathbf{A}$$

При численном расчете на месте нулей могут быть очень маленькие значения, в том числе и отрицательные, что обусловлено вычислительными ошибками, которыми можно пренебречь.

Метод Гивенса

В основе метода Гивенса (метод вращений) лежит последовательность из нескольких поворотов Гивенса (линейных преобразований матрицы **A**), которые, например для $n = 3$ можно записать в виде:

$$\mathbf{G}_{22} = \begin{bmatrix} \cos(\theta) & 0 & -\sin(\theta) \\ 0 & 1 & 0 \\ \sin(\theta) & 0 & \cos(\theta) \end{bmatrix}$$

Формируя определенным образом последовательность подобных поворотов можно вычислить требуемые матрицы **Q** и **R** следующим образом:

$$\mathbf{Q} = \mathbf{G}_{1,2}^{\top} \cdots \mathbf{G}_{1,n}^{\top} \cdots \mathbf{G}_{2,3}^{\top} \cdots \mathbf{G}_{2,n}^{\top} \cdots \mathbf{G}_{n-2,n}^{\top} \mathbf{G}_{n-1,n}^{\top}$$

$$\mathbf{R} = \mathbf{G}_{n-1,n} \mathbf{G}_{n-2,n} \mathbf{G}_{n-2,n-1} \cdots \mathbf{G}_{1,3} \mathbf{G}_{1,2} \mathbf{A}$$

Последовательность поворотов формируется при $i = 1, \ldots, n-1$ и $j = i+1, \ldots, n$ в ходе итерационного процесса.

Инициализация:

$$\mathbf{Q} = \mathbf{I}$$
$$\mathbf{R} = \mathbf{A}$$

На каждой итерации соответствующей пары (i, j) вычисляем:

$$\mathbf{G}_{n \times n} = \mathbf{I}_{n \times n}$$
$$g_{ii} = g_{jj} = \frac{a_{ii}}{\sqrt{a_{ii}^2 + a_{ji}^2}};$$
$$g_{ij} = \frac{a_{ji}}{\sqrt{a_{ii}^2 + a_{ji}^2}};$$
$$g_{ji} = -\frac{a_{ji}}{\sqrt{a_{ii}^2 + a_{ji}^2}};$$
$$\mathbf{Q} = \mathbf{Q}\,\mathbf{G}^\top$$
$$\mathbf{A} = \mathbf{G}\,\mathbf{R}$$
$$\mathbf{R} = \mathbf{A}$$

Используется для эффективного вычисление обратной матрицы:

$$\mathbf{A}^{-1} = \left(\mathbf{QR}\right)^{-1} = \mathbf{R}^{-1}\mathbf{Q}^{-1} = \mathbf{R}^{-1}\mathbf{Q}^{\top}$$

Обратную матрицу $\mathbf{R}^{-1}$ легко считать, т.к. она треугольная, что существенно уменьшает количество операций при вычислении обратной матрицы $\mathbf{A}^{-1}$.

`eig` - спектральное разложение;
`eigs` - собственные значения и собственные вектора матрицы;
`svd` - сингулярное разложение;
`svds` - сингулярные значения и сингулярные вектора матрицы;
`lu` - *LU*-разложение, *LUP*-разложение;
`chol` - *LL*-разложение;
`ldl` - *LDL*-разложение;
`schur` - разложение Шура;
`qr` - *QR*-разложение;

Система уравнений:

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + \ldots + a_{1m}x_m = b_1 \\ a_{11}x_1 + a_{12}x_2 + \ldots + a_{1m}x_m = b_1 \\ a_{21}x_1 + a_{22}x_2 + \ldots + a_{2m}x_m = b_2 \\ \ldots \\ a_{n1}x_1 + a_{n2}x_2 + \ldots + a_{nm}x_m = b_n \end{cases}$$

где:

- n - количество уравнений;
- m - количество неизвестных;
- x_i - неизвестные;
- a_{ij} - коэффициенты;
- b_i - свободные члены.

В матричной форме:

$$\mathbf{A}\,\mathbf{x} = \mathbf{b}$$

Решение имеет вид:

$$\mathbf{x} = \mathbf{A}^{-1}\,\mathbf{b}$$

где:

- $\mathbf{A}$ - матрица $n \times m$;
- $\mathbf{x}$ - вектор $m \times 1$;
- $\mathbf{b}$ - вектор $n \times 1$;

Свойства систем линейных алгебраических уравнений:

- если все $b_i = 0$ - однородная система, иначе неоднородная;
- если $m = n$ - совместная система (единственное решение);
- если $n > m$ - переопределенная система (множество решений);
- если $n < m$ - недоопределенная система (нет решений);

Теорема Кронекера-Капелли:

Система уравнений $\mathbf{A}\,\mathbf{x} = \mathbf{b}$ разрешима (имеет хотя бы одно решение) тогда и только тогда, когда $\mathrm{rang}(\mathbf{A}) = \mathrm{rang}(\mathbf{A}, \mathbf{b})$, где $(\mathbf{A}, \mathbf{b})$ - расширенная матрица системы. В этом случае количество искомых переменных равно порядку матрицы, а решение системы будет единственным.

Прямые:

(Непосредственное вычисление решения за конечное количество операций)

- метод решения через $\mathbf{A}^{-1}$;
- метод Крамера;
- метод Гаусса;
- метод Гаусса-Жордана;
- разложение Холецкого;
- и др.

Итерационные:

(Поиск решения в результате последовательных приближений)

- метод Якоби;
  (метод простой итерации)
- метод Гаусса-Зейделя;
- метод релаксации;
- и др.

Метод через $\mathbf{A}^{-1}$

Метод через $\mathbf{A}^{-1}$ - это метод с непосредственным вычислением обратной матрицы, общий алгоритм которого может быть представлен в виде:

- вычисляем алгебраические дополжения:

$$A_{ij} = (-1)^{i+j} \bar{M}_{ij}$$

- вычисляем обратную матрицу

$$\mathbf{A}^{-1} = \frac{1}{\det(\mathbf{A})} \operatorname{adj} \mathbf{C}$$

- находим значения вектора $\mathbf{x}$

$$\mathbf{x} = \mathbf{A}^{-1} \mathbf{b}$$

Условия применимости:

- $\det(\mathbf{A}) \neq 0$.

Алгоритм требует колоссальных вычислительных ресурсов, если не использовать разложение исходной матрицы $\mathbf{A}$.

Метод Крамера

Метод Крамера - это метод решения СЛАУ в котором каждый элемент вектора решения определяется следующим образом:

$$x_i = \frac{1}{\det(\mathbf{A})} \begin{vmatrix} a_{1,1} & \dots & a_{1,i-1} & b_1 & a_{1,i+1} & \dots & a_{1,n} \\ & & & \vdots & & & \\ a_{n,1} & \dots & a_{n,i-1} & b_n & a_{n,i+1} & \dots & a_{n,n} \end{vmatrix}$$

Условия применимости:

- $\mathbf{A}$ - квадратная матрица;
- $\det(\mathbf{A}) \neq 0$.

Метод эффективен для получения аналитического решения для СЛАУ небольших размерностей. Численная интерпретация требует существенных вычислительных ресурсов, особенно при увеличении порядка системы.

Метод Гаусса

Метод Гаусса (метод исключения переменных) - это прямой метод решения СЛАУ, в котором получение решения состоит из двух этапов:

- прямой ход: методом последовательного исключения переменных приводим **A** к ступенчатому виду;
- обратный ход: из последнего не нулевого уравнения находим значение $(x_{j1} \dots x_{jr})$ и подставляем в предыдущее уравнения.

Условия применимости:

- **A** - квадратная матрица.

Метод используется для решения большого числа задач, однако при увеличении порядка системы возрастают требования к вычислительной мощности.

Устойчивость метода сильно зависит от диагональных элементов матрицы **A**. Если есть нулевые диагональные элементы необходимо осуществить элементарные преобразования над строками/столбцами для исключения этой ситуации.

Алгоритм прямого хода:

- $i = 1$;
- $k = i + 1$;
- $A(k,:) = A(k,:) - A(i,:)\,\dfrac{A(k,i)}{A(i,i)}$;
- $b(k) = b(k) - b(i)\,\dfrac{A(k,i)}{A(i,i)}$;
- $k = k + 1$ - по всем строкам для каждого i;
- $i = i + 1$ - и повторяем процедуру.

Алгоритм обратного хода:

- $i = n$;
- $x_i = \dfrac{b(i) - \sum\limits_{k=i+1}^{n} A(i,k)\,x(k)}{A(i,i)}$;
- $i = i - 1$ - и повторяем процедуру.

Исходные данные:

$$\mathbf{A} = \begin{bmatrix} 1 & 2 & 3 \\ 3 & 1 & 2 \\ 2 & 3 & 1 \end{bmatrix}, \qquad \mathbf{b} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$$

Итерация $i = 1$, $k = i + 1 = 2$:

$$\begin{bmatrix} 3 & 1 & 2 \end{bmatrix} - \begin{bmatrix} 1 & 2 & 3 \end{bmatrix} \cdot \frac{3}{1} = \begin{bmatrix} 0 & -5 & -7 \end{bmatrix}$$

$$\begin{bmatrix} 1 \end{bmatrix} - \begin{bmatrix} 1 \end{bmatrix} \cdot \frac{3}{1} = \begin{bmatrix} -2 \end{bmatrix}$$

Результат итерации:

$$\mathbf{A} = \begin{bmatrix} 1 & 2 & 3 \\ 0 & -5 & -7 \\ 2 & 3 & 1 \end{bmatrix}, \qquad \mathbf{b} = \begin{bmatrix} 1 \\ -2 \\ 1 \end{bmatrix}$$

На текущий момент:

$$\mathbf{A} = \begin{bmatrix} 1 & 2 & 3 \\ 0 & -5 & -7 \\ 2 & 3 & 1 \end{bmatrix}, \qquad \mathbf{b} = \begin{bmatrix} 1 \\ -2 \\ 1 \end{bmatrix}$$

Итерация $i = 1$, $k = k + 1 = 2 + 1 = 3$:

$$\begin{bmatrix} 2 & 3 & 1 \end{bmatrix} - \begin{bmatrix} 1 & 2 & 3 \end{bmatrix} \cdot \frac{2}{1} = \begin{bmatrix} 0 & -1 & -5 \end{bmatrix}$$

$$\begin{bmatrix} 1 \end{bmatrix} - \begin{bmatrix} 1 \end{bmatrix} \cdot \frac{2}{1} = \begin{bmatrix} -1 \end{bmatrix}$$

Результат итерации:

$$\mathbf{A} = \begin{bmatrix} 1 & 2 & 3 \\ 0 & -5 & -7 \\ 0 & -1 & -5 \end{bmatrix}, \qquad \mathbf{b} = \begin{bmatrix} 1 \\ -2 \\ -1 \end{bmatrix}$$

На текущий момент:

$$\mathbf{A} = \begin{bmatrix} 1 & 2 & 3 \\ 0 & -5 & -7 \\ 0 & -1 & -5 \end{bmatrix}, \qquad \mathbf{b} = \begin{bmatrix} 1 \\ -2 \\ -1 \end{bmatrix}$$

Итерация $i = 2$, $k = i + 1 = 3$:

$$\begin{bmatrix} 0 & -1 & -5 \end{bmatrix} - \begin{bmatrix} 0 & -5 & -7 \end{bmatrix} \cdot \frac{-1}{-5} = \begin{bmatrix} 0 & 0 & -3.6 \end{bmatrix}$$

$$\begin{bmatrix} -1 \end{bmatrix} - \begin{bmatrix} (-2) \end{bmatrix} \cdot \frac{-1}{-5} = \begin{bmatrix} -0.6 \end{bmatrix}$$

Результат итерации:

$$\mathbf{A} = \begin{bmatrix} 1 & 2 & 3 \\ 0 & -5 & -7 \\ 0 & 0 & -3.6 \end{bmatrix}, \qquad \mathbf{b} = \begin{bmatrix} 1 \\ -2 \\ -0.6 \end{bmatrix}$$

После завершения прямого хода:

$$\mathbf{A} = \begin{bmatrix} 1 & 2 & 3 \\ 0 & -5 & -7 \\ 0 & 0 & -3.6 \end{bmatrix}, \qquad \mathbf{b} = \begin{bmatrix} 1 \\ -2 \\ -0.6 \end{bmatrix}$$

Обратный ход:

$$x_3 = -0.6/-3.6 = 0.1667$$
$$x_2 = \frac{-2-(-7\cdot 0.1667)}{-5} = 0.1666$$
$$x_1 = \frac{1-(2\cdot 0.1666+3\cdot 0.1667)}{1} = 0.1667$$

Полученное решение:

$$\mathbf{x} = \begin{bmatrix} 0.1667 & 0.1666 & 0.1667 \end{bmatrix}^\top$$

Метод Гаусса-Жордана

Метод Гаусса-Жордана - это численный метод решения СЛАУ, который, по своей сути, похож на жметод Гаусса, но за исключением того, что матрица **A** и вектор **b** объединяются в одну расширенную матрицу **C**:

$$\mathbf{C} = (\mathbf{A}, \mathbf{b})$$

При использовании подобного подхода, несколько меняется суть обратного хода. Если в методе Гаусса при обратном ходе непосредственно вычисляются элементы вектора **x**, то в метода Гаусса-Жордана на обратном ходе матрица **C** приводится к диагональному виду (не считая последнего столбца).

Если в ходе решения приводить матрицу **C** к единичной, тогда решение - вектор **x** будет находится в ее последнем столбце.

Условие применимости:

- **A** - квадратная матрица.

Алгоритм прямого хода:

- $i = 1$;
- $k = i + 1$;
- $C(k,:) = C(k,:) - C(i,:)\dfrac{C(k,i)}{C(i,i)}$;
- $k = k + 1$ - по всем строкам для каждого i;
- $i = i + 1$ - и повторяем процедуру.

Алгоритм обратного хода:

- $i = n$;
- $k = i - 1$;
- $C(k,:) = C(k,:) - C(i,:)\dfrac{C(k,i)}{C(i,i)}$;
- $k = k - 1$ - по всем строкам для каждого i;
- $i = i - 1$ - и повторяем процедуру.

После завершения прямого хода:

$$= \begin{bmatrix} 1 & 2 & 3 & 1 \\ 0 & -5 & -7 & -2 \\ 0 & 0 & -3.6 & -0.6 \end{bmatrix}$$

Итерация $i = 3$, $k = i - 1 = 2$:

$$\begin{bmatrix} 0 & -5 & -7 & -2 \end{bmatrix} - \begin{bmatrix} 0 & 0 & -3.6 & -0.6 \end{bmatrix} \cdot \frac{-7}{-3.6} = \begin{bmatrix} 0 & -5 & 0 & -0.8333 \end{bmatrix}$$

Результат итерации:

$$= \begin{bmatrix} 1 & 2 & 3 & 1 \\ 0 & -5 & 0 & -0.8333 \\ 0 & 0 & -3.6 & -0.6 \end{bmatrix}$$

На текущий момент:

$$= \begin{bmatrix} 1 & 2 & 3 & 1 \\ 0 & -5 & 0 & -0.8333 \\ 0 & 0 & -3.6 & -0.6 \end{bmatrix}$$

Итерация $i = 3$, $k = k - 1 = 1$:

$$\begin{bmatrix} 1 & 2 & 3 & 1 \end{bmatrix} - \begin{bmatrix} 0 & 0 & -3.6 & -0.6 \end{bmatrix} \cdot \frac{3}{-3.6} = \begin{bmatrix} 1 & 2 & 0 & 0.5 \end{bmatrix}$$

Результат итерации:

$$= \begin{bmatrix} 1 & 2 & 0 & 0.5 \\ 0 & -5 & 0 & -0.8333 \\ 0 & 0 & -3.6 & -0.6 \end{bmatrix}$$

На текущий момент:

$$= \begin{bmatrix} 1 & 2 & 0 & 0.5 \\ 0 & -5 & 0 & -0.8333 \\ 0 & 0 & -3.6 & -0.6 \end{bmatrix}$$

Итерация $i = 2$, $k = i - 1 = 1$:

$$\begin{bmatrix} 1 & 2 & 0 & 0.5 \end{bmatrix} - \begin{bmatrix} 0 & -5 & 0 & -0.8333 \end{bmatrix} \cdot \frac{2}{-5} = \begin{bmatrix} 1 & 0 & 0 & 0.168 \end{bmatrix}$$

Результат итерации:

$$= \begin{bmatrix} 1 & 0 & 0 & 0.168 \\ 0 & -5 & 0 & -0.8333 \\ 0 & 0 & -3.6 & -0.6 \end{bmatrix}$$

После завершения обратного хода:

$$= \begin{bmatrix} 1 & 0 & 0 & 0.168 \\ 0 & -5 & 0 & -0.8333 \\ 0 & 0 & -3.6 & -0.6 \end{bmatrix}$$

Обратный ход:

$$x_1 = 0.168/1 = 0.1680$$
$$x_2 = -0.8333/-5 = 0.1660$$
$$x_3 = -0.6/-3.6 = 0.1667$$

Полученное решение:

$$\mathbf{x} = \begin{bmatrix} 0.1680 \\ 0.1660 \\ 0.1667 \end{bmatrix}$$

Метод разложения Холецкого

Метод разложения Холецкого - это метод решения СЛАУ, в котором сначала определяется разложение Холецкого матрицы $\mathbf{A}$, т.е. матрица $\mathbf{L}$, а затем вычисляется решение $\mathbf{x}$ последовательным решением двух уравнений:

$$\mathbf{L}\,\mathbf{y} = \mathbf{b}$$
$$\mathbf{L}^{\top}\,\mathbf{x} = \mathbf{y}$$

Поскольку матрица $\mathbf{L}$ треугольная, то, по сути, механизм решения аналогичен обратному ходу метода Гаусса, только для первого уравнения начинаем считать с первой, а не последней строки.

Условие применимости:

- $\mathbf{A}$ - положительно-определенная матрицы.

Несимметричность матрицы $\mathbf{A}$ не накладывает ограничений на применимость метода, но полученное решение будет не соответствовать истинному.

Аналогичный подход может быть использован и с использованием LU-разложения

$$\mathbf{A} = \begin{bmatrix} 1.4 & 1 & 1 \\ 1 & 0.9 & 1 \\ 1 & 1 & 1.4 \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}, \quad \mathbf{L} = \begin{bmatrix} 1.18 & 0 & 0 \\ 0.85 & 0.43 & 0 \\ 0.85 & 0.66 & 0.5 \end{bmatrix}$$

Решаем $\mathbf{L}\,\mathbf{y} = \mathbf{b}$:

$$y_1 = 1/1.18 = 0.85$$
$$y_2 = \frac{1 - (0.85 \cdot 0.85)}{0.43} = 0.65$$
$$y_3 = \frac{1 - (0.85 \cdot 0.85 + 0.65 \cdot 0.66)}{0.5} = -0.31$$

Решаем $\mathbf{L}^\top \mathbf{x} = \mathbf{y}$:

$$x_3 = -0.31/0.5 = -0.62$$
$$x_2 = \frac{0.65 - 0.66 \cdot (-0.62)}{0.43} = 2.48$$
$$x_1 = \frac{0.85 - (0.85 \cdot (-0.62) + 0.85 \cdot 2.48)}{1.18} = -0.62$$

Итерационные методы

Итерационные методы решения СЛАУ строятся на идее последовательного приближения к истинному решению (где оно достижимо).

Основные подходы следующие:

- основанные на расщеплении:

$$(\mathbf{M}-\mathbf{N})\,\mathbf{x}=\mathbf{b}\Leftrightarrow\mathbf{M}\,\mathbf{x}=\mathbf{N}\,\mathbf{x}+\mathbf{b}\Rightarrow\mathbf{M}\,\mathbf{x}_{k+1}=\mathbf{N}\,\mathbf{x}_k+\mathbf{b}$$

(схема вычислений: «следующий через предыдущий»)

- вариационные:

$$\mathbf{A}\,\mathbf{x}=\mathbf{b}\Rightarrow\ \|\mathbf{A}\,\mathbf{x}-\mathbf{b}\|\ \to\min$$

(как правило, речь идет о применении метод оптимизации, например, метода наименьших квадратов, для решения переопределенных задач)

- проекционные:

$$\mathbf{A}\,\mathbf{x}=\mathbf{b}\Rightarrow(\mathbf{A}\,\mathbf{x},\mathbf{m})=(\mathbf{b},\mathbf{m})\,\forall\,\mathbf{m}$$

(в основе этих методов проектирование неизвестного вектора на некоторое линейное пространство, формируемое из известных данных)

Метод Якоби

Метод Якоби (метод простой итерации) - итерационный метод решения СЛАУ, в котором итерационная схема строится следующим образом:

$$\mathbf{x}_{k+1} = \mathbf{B}\,\mathbf{x}_k + \mathbf{g}$$

или в поэлементном виде:

$$x_{i,\,k+1} = \frac{1}{a_{ii}} \left(b_i - \sum_{j \neq i} a_{ij}\, x_{j,\,k} \right)$$

Привести систему к итерационному виду можно несколькими способами:

- $\mathbf{B} = \mathbf{I} - \mathbf{D}^{-1}\mathbf{A} = \mathbf{D}^{-1}(\mathbf{D} - \mathbf{A}),\ \mathbf{g} = \mathbf{D}^{-1}\,\mathbf{b}$;
- $\mathbf{B} = -\mathbf{D}^{-1}(\mathbf{L} + \mathbf{U}) = -\mathbf{D}^{-1}(\mathbf{A} - \mathbf{D}),\ \mathbf{g} = \mathbf{D}^{-1}\,\mathbf{b}$;
- $D_{ii}^{-1} = 1/D_{ii},\ D_{ii} \neq 0,\ i = 1, 2 \ldots n$;

где:

$\mathbf{D} = \mathrm{diag}(\mathbf{A})$, $\mathbf{L}, \mathbf{U}$ - верхняя и нижняя треугольные части $\mathbf{A}$.

Начальные условия $\mathbf{x}_0$ произвольные, но их выбор влияет на сходимость.

Условие сходимости:

Решение методом Якоби сходится (для произвольных начальных условий) если:

$$|\lambda_i(\mathbf{B})| < 1, \quad \text{где} \quad \mathbf{B} = -\mathbf{D}^{-1}(\mathbf{L}+\mathbf{U})$$

При условии сходимости справедливы следующие утверждения:

- $\|\mathbf{B}\| < 1$ при $\forall\, \mathbf{x}_0$;
- $q = \|\mathbf{B}\|$ со скоростью геометрической прогрессии;
- $\|\mathbf{x}_k - \mathbf{x}\| < q^k \|\mathbf{x}_0 - \mathbf{x}\|$.

Критерии остановки:

- $k \geqslant k_{\max}$;
- $\|\mathbf{x}_k - \mathbf{x}_{k-1}\| \leqslant \varepsilon$;
  (по сути, идет проверка на относительную ошибку вычислений, может ложно срабатывать при медленной сходимости метода)
- $\|\mathbf{A}\,\mathbf{x}_k - \mathbf{b}\| \leqslant \varepsilon$.
  (по сути, идет проверка на абсолютную ошибку вычислений, требует умножения матрицы на вектор, а как следствие требует доволнительных вычислительных ресурсов)

Схема вычислений:

$$\mathbf{x}_0 \xrightarrow[i=1,\dots,n]{} \mathbf{x}_1 \xrightarrow[k=1]{\mathbf{x}_k = \mathbf{x}_{k+1}} \mathbf{x}_1 \xrightarrow[i=1,\dots,n]{} \mathbf{x}_2 \xrightarrow[k=2]{\mathbf{x}_k = \mathbf{x}_{k+1}} \mathbf{x}_2 \xrightarrow[i=1,\dots,n]{} \mathbf{x}_3 \to \dots$$

Алгоритм:

- $k = 0$;
- $\mathbf{x}_k = [0]$;
- проверяем критерии остановки;
- вычисляем i элемент $x_{i,\,k+1} = \dots$;
- закончив вычислять $\mathbf{x}_{k+1}$, перезаписываем: $\mathbf{x}_k = \mathbf{x}_{k+1}$;
- $k = k + 1$;
- повторяем процесс вычисления $x_{i,\,k+1} = \dots$.

Метод Гаусса-Зейделя

Метод Гаусса-Зейделя - итерационный метод решения СЛАУ, в котором итерационная схема строится следующим образом:

$$(\mathbf{L} + \mathbf{D})\,\mathbf{x}_{k+1} = -\,\mathbf{U}\,\mathbf{x}_k + \mathbf{B}$$

где:

$\mathbf{D} = \mathrm{diag}\,(\mathbf{A})$ - диагональная матрица;
$\mathbf{L}$, $\mathbf{U}$ - нижняя и верхняя треугольные части $\mathbf{A}$.

или в поэлементном виде:

$$x_{i,\,k+1} = \sum_{j=1}^{i-1} c_{ij}\,x_{j,\,k+1} + \sum_{j=i+1}^{n} c_{ij}\,x_{j,\,k} + d_i$$

где

$$c_{ij} = \begin{cases} -\dfrac{a_{ij}}{a_{ii}} & j \neq i \\ 0 & j = i \end{cases}, \qquad d_i = \frac{b_i}{a_{ii}}$$

Условие сходимости:

Решение методом Гаусса-Зейделя сходится при выполнении условия:

$$\|\mathbf{P}\| < 1, \quad \text{где} \quad \mathbf{P} = -(\mathbf{L} + \mathbf{D})^{-1}\,\mathbf{U}$$

Как правило сходится для:

- симметричных и положительно определенных матриц;
- матриц с доминимирующими диагональным элементами.

Критерии остановки:

- $k \geqslant k_{\max}$;
- $\|\mathbf{A}\,\mathbf{x}_k - \mathbf{b}\| \leqslant \varepsilon$.

Может сходить и при невыполнении этих условий

Схема вычислений (используется один вектор):

$$\begin{bmatrix} x_{1,\,k} \\ x_{2,\,k} \\ x_{3,\,k} \end{bmatrix} \xrightarrow[i=1]{} \begin{bmatrix} x_{1,\,k+1} \\ x_{2,\,k} \\ x_{3,\,k} \end{bmatrix} \xrightarrow[i=2]{} \begin{bmatrix} x_{1,\,k+1} \\ x_{2,\,k+1} \\ x_{3,\,k} \end{bmatrix} \xrightarrow[i=3]{} \begin{bmatrix} x_{1,\,k+1} \\ x_{2,\,k+1} \\ x_{3,\,k+1} \end{bmatrix} \xrightarrow{k=k+1} \begin{bmatrix} x_{1,\,k} \\ x_{2,\,k} \\ x_{3,\,k} \end{bmatrix} \to \ldots$$

Алгоритм (используется один вектор):

- $k = 0$;
- $\mathbf{x}_k = [0]$;
- проверяем критерии остановки;
- вычисляем i элемент $x_{i,\,k+1} = \ldots$;
- закончив вычислять $\mathbf{x}_{k+1}$, ставим $i = 1$;
- $k = k + 1$;
- повторяем процесс вычисления $x_{i,\,k+1} = \ldots$.

Метод релаксации

Метод релаксации - это усовершенствование метода Гаусса-Зейделя, обеспечивающее более быструю сходимость, в котором итерационный процесс имеет вид:

$$\left(\omega\,\mathbf{L}+\mathbf{D}\right)\,\mathbf{x}_{k+1}=-\left(\omega\,\mathbf{U}+(\omega-1)\,\mathbf{D}\right)\,\mathbf{x}_{k}+\omega\,\mathbf{B}\,,$$

где:

ω - коэффициент релаксации.

и в поэлементном виде:

$$x_{i,\,k+1}=(1-\omega)\,x_{i,\,k}+\frac{\omega}{a_{ii}}\left(b_i-\sum_{j<i}a_{ij}\,x_{j,\,k+1}-\sum_{j>i}a_{ij}\,x_{j,\,k}\right)$$

Выбор ω не тривиален. Для симметричных, положительно полуопределенных матриц, при $0 < \omega < 2$ метод гарантированно сходится, а оптимальное значение ω определяется следующим образом:

$$\omega = 1 + \left(\frac{\rho(\mathbf{T})}{1 + \sqrt{1 - \rho(\mathbf{T})^2}} \right)^2$$

где:

$\mathbf{T} = \mathbf{I} - \mathbf{D}^{-1}\mathbf{A}$ - зависит от системы;
$\rho(\cdot)$ - спектральный радиус матрицы.

Спектральный радиус матрицы:

$$\rho(\mathbf{T}) = \max \left| \lambda_i(\mathbf{T}) \right|$$

Метод сопряженных градиентов

Метод сопряженных градиентов - один из итерационных методов, так называемого Крыловского типа - группы проекционных методов. Решение СЛАУ получается через решение задачи оптимизации с минимизацией:

$$(\mathbf{A}\,\mathbf{x}, \mathbf{x}) - 2\,(\mathbf{b}, \mathbf{x}) \to \min$$

решение которой аналитически сформулировано в виде алгоритма.

Условия сходимости:

- $\mathbf{A}$ - действительная, симметричная и положительно определенная.

Критерии остановки:

- $k \geqslant k_{\max}$ (должно завершаться за n итераций);
- $\dfrac{\|\mathbf{r}_k\|}{\|\mathbf{b}\|} \leqslant \varepsilon$.
- $\|\mathbf{x}_k - \mathbf{x}_{k-1}\| \leqslant \varepsilon$;
- $\|\mathbf{A}\,\mathbf{x}_k - \mathbf{b}\| \leqslant \varepsilon$.

Инициализация:

$$\mathbf{r}_0 = \mathbf{b} - \mathbf{A}\,\mathbf{x}_0$$
$$\mathbf{z}_0 = \mathbf{r}_0$$

***k*-ая итерация:**

$$\alpha_k = \frac{(\mathbf{r}_{k-1}, \mathbf{r}_{k-1})}{(\mathbf{A}\,\mathbf{z}_{k-1}, \mathbf{z}_{k-1})}$$
$$\mathbf{x}_k = \mathbf{x}_{k-1} + \alpha_k\,\mathbf{z}_{k-1}$$
$$\mathbf{r}_k = \mathbf{r}_{k-1} - \alpha_k\,\mathbf{A}\,\mathbf{z}_{k-1}$$
$$\beta_k = \frac{(\mathbf{r}_k, \mathbf{r}_k)}{(\mathbf{r}_{k-1}, \mathbf{r}_{k-1})}$$
$$\mathbf{z}_k = \mathbf{r}_k + \beta_k\,\mathbf{z}_{k-1}$$

Для $\mathbf{A}$ с комплексными числами есть соответствующий вариант алгоритма, а требования к сходимости дополнительно включают самосопряженность матрицы $\mathbf{A}$

Метод бисопряженных градиентов

Метод бисопряженных градиентов - один из итерационных методов, так называемого Крыловского типа - группы проекционных методов, который является обобщением метода сопряженных градиентов на случай несимметричной матрицы **A**.

Условия сходимости:

- **A** - действительная и положительно определенная.

Критерии остановки:

- $k \geqslant k_{\max}$ (должно завершаться за n итераций);
- $\dfrac{\|\mathbf{r}_k\|}{\|\mathbf{b}\|} \leqslant \varepsilon$.
- $\|\mathbf{x}_k - \mathbf{x}_{k-1}\| \leqslant \varepsilon$;
- $\|\mathbf{A}\,\mathbf{x}_k - \mathbf{b}\| \leqslant \varepsilon$.

Инициализация:

$$\mathbf{r}_0 = \mathbf{b} - \mathbf{A}\,\mathbf{x}_0$$
$$\mathbf{p}_0 = \mathbf{r}_0$$
$$\mathbf{z}_0 = \mathbf{r}_0$$
$$\mathbf{s}_0 = \mathbf{r}_0$$

***k*-ая итерация:**

$$\alpha_k = \frac{(\mathbf{p}_{k-1}, \mathbf{r}_{k-1})}{(\mathbf{s}_{k-1}, \mathbf{A}\,\mathbf{z}_{k-1})}$$
$$\mathbf{x}_k = \mathbf{x}_{k-1} + \alpha_k\,\mathbf{z}_{k-1}$$
$$\mathbf{r}_k = \mathbf{r}_{k-1} - \alpha_k\,\mathbf{A}\,\mathbf{z}_{k-1}$$
$$\mathbf{p}_k = \mathbf{p}_{k-1} - \alpha_k\,\mathbf{A}^\top\,\mathbf{s}_{k-1}$$
$$\beta_k = \frac{(\mathbf{p}_k, \mathbf{r}_k)}{(\mathbf{p}_{k-1}, \mathbf{r}_{k-1})}$$
$$\mathbf{z}_k = \mathbf{r}_k + \beta_k\,\mathbf{z}_{k-1}$$
$$\mathbf{s}_k = \mathbf{p}_k + \beta_k\,\mathbf{s}_{k-1}$$

Стабилизированный метод бисопряженных градиентов

Стабилизированный метод бисопряженных градиентов - один из итерационных методов, так называемого Крыловского типа - группы проекционных методов, который является обобщением метода бисопряженных градиентов на случай несимметричной матрицы $\mathbf{A}$, но при этом обладает вычислительной устойчивостью.

Условия сходимости:

- $\mathbf{A}$ - действительная и положительно определенная.

Критерии остановки:

- $k \geqslant k_{\max}$ (должно завершаться за n итераций);
- $\dfrac{\|\mathbf{r}_k\|}{\|\mathbf{b}\|} \leqslant \varepsilon$.
- $\|\mathbf{x}_k - \mathbf{x}_{k-1}\| \leqslant \varepsilon$;
- $\|\mathbf{A}\,\mathbf{x}_k - \mathbf{b}\| \leqslant \varepsilon$.

Инициализация:

$$\mathbf{r}_0 = \mathbf{b} - \mathbf{A}\,\mathbf{x}_0$$
$$\widetilde{\mathbf{r}} = \mathbf{r}_0$$
$$\rho_0 = \alpha_0 = \omega_0 = 1$$
$$\upsilon_0 = \mathbf{p}_0 = 0$$

Скалярные произведения:

$$(\mathbf{q}, \mathbf{d}) = \sum_{i=1}^{n} q_i^* \, d_i$$

$$[\mathbf{q}, \mathbf{d}] = \sum_{i=1}^{n} q_i \, d_i$$

***k*-ая итерация:**

$$\rho_k = (\widetilde{\mathbf{r}}, \mathbf{r}_{k-1})$$
$$\beta_k = \frac{\rho_k}{\rho_{k-1}} \, \frac{\alpha_{k-1}}{\omega_{k-1}}$$
$$\mathbf{p}_k = \mathbf{r}_{k-1} + \beta_k \left(\mathbf{p}_{k-1} - \omega_{k-1}\,\upsilon_{k-1}\right)$$
$$\upsilon_k = \mathbf{A}\,\mathbf{p}_k$$
$$\alpha_k = \frac{\rho_k}{(\widetilde{\mathbf{r}}, \upsilon_k)}$$
$$\mathbf{s}_k = \mathbf{r}_{k-1} - \alpha_k\,\upsilon_k$$
$$\mathbf{t}_k = \mathbf{A}\,\mathbf{s}_k$$
$$\omega_k = \frac{[\mathbf{t}_k, \mathbf{s}_k]}{[\mathbf{t}_k, \mathbf{t}_k]}$$
$$\mathbf{x}_k = \mathbf{x}_{k-1} + \alpha_k\,\mathbf{p}_k + \omega_k\,\mathbf{s}_k$$
$$\mathbf{r}_k = \mathbf{s}_k - \omega_k\,\mathbf{t}_k$$

`cgs` - метод сопряженных градиентов;
`bicg` - метод бисопряженных градиентов;
`bicgstab` - стабилизированный метод бисопряженных градиентов;
`linsolve` - численное решение СЛАУ.

Интерполяция

Интерполяция - это вычисление промежуточных значений функции $\mathbf{f}(\mathbf{x})$ на сетке значений аргумента $\mathbf{x}$ для заданного дискретного набора $\{\mathbf{y}, \mathbf{f}(\mathbf{y})\}$ значений этой функции, где $\mathbf{x} \in [a, b]$ и $\mathbf{y} \in [a, b]$, а интервал $[a, b]$ определен как $[\min(\mathbf{y}), \max(\mathbf{y})]$.

Экстраполяция

Экстраполяция - это вычисление промежуточных значений функции $\mathbf{f}(\mathbf{x})$ на сетке значений аргумента $\mathbf{x}$ для заданного дискретного набора $\{\mathbf{y}, \mathbf{f}(\mathbf{y})\}$ значений этой функции, где $\mathbf{x} \notin [a, b]$ и $\mathbf{y} \in [a, b]$, а интервал $[a, b]$ определен как $[\min(\mathbf{y}), \max(\mathbf{y})]$.

Существует множество методов интерполяции для одномерных и многомерных случаев и т.п. Основная классификация следующая:

- интерполяция методом ближайшего соседа;
- полиномиальная интерполяция:
  - линейная интерполяция;
  - интерполяция полиномом Лагранжа;
  - интерполяция полиномом Ньютона вперед;
  - интерполяция полиномом Ньютона назад;
  - интерполяция кубическими сплайнами;
- интерполяция функции нескольких переменных:
  - билинейная интерполяция;
  - бикубическая интерполяция.
- рациональная интерполяция.

Для каждого x_i:

$$\Delta_1 = \left| x_i - y_k \right|$$
$$\Delta_2 = \left| x_i - y_{k+1} \right|$$

Как только $\Delta_2 \leqslant \Delta_1$:

$$k = k + 1$$

Значение функции:

$$f(x_i) = f(y_k)$$

Интерполяция одномерной функции

При полиномиальной интерполяции функция $f(x)$ на интервале $[y_k, y_{k+1}]$ (локально) или на интервале $[a, b]$ (глобально) аппроксимируется полиномом $p(x)$.

Линейная интерполяция

Функция $f(x)$ на интервале $[y_k, y_{k+1}]$ аппроксимируется прямой (полиномом первой степени), т.е.:

$$\frac{f(x) - f(y_k)}{f(y_{k+1}) - f(y_k)} = \frac{x - y_k}{y_{k+1} - y_k}$$

Значения функции $f(x)$ определяются следующим образом:

$$f(x) \approx f(y_k) + \frac{f(y_{k+1}) - f(y_k)}{y_{k+1} - y_k} \left(x - y_k \right)$$

Переход к $k = k + 1$ происходит при $x \geq y_{k+1}$.

Интерполяция полиномом Лагранжа

Интерполяция полиномом Лагранжа заключается в том, что функция $f(x)$ на интервале $[a, b]$ для $n + 1$ точек дискретного набора $\{\mathbf{y}, \mathbf{f}(\mathbf{y})\}$ аппроксимируется единственным полиномом Лагранжа порядка не выше n, для которого:

$$L(x = y_k) = f(y_k)$$

Сам полином Лагранжа определен следующим образом:

$$L(x) = \sum_{k=0}^{n} f(y_k)\, l_k(x)\,,$$

где $l_k(x)$ - базисный полином, вычисляемый по формуле:

$$l_k(x) = \prod_{i=0,\, i \neq k}^{n} \frac{x - y_i}{y_k - y_i}$$

Пусть заданна таблично $f(x) = x^2$:

k	0	1	2
y	1	2	3
$f(y)$	1	4	9

Полином Лагранжа второго порядка:

$$L(x) = 1 \cdot \frac{(x-2)(x-3)}{(1-2)(1-3)} + 4 \cdot \frac{(x-1)(x-3)}{(2-1)(2-3)} + 9 \cdot \frac{(x-1)(x-2)}{(3-1)(3-2)}$$
$$= x^2$$

В результате аналитических вычислений получили точную аппроксимацию (порядки $L(x)$ и $f(x)$ совпали).

Пусть заданна таблично $f(x) = x^3$:

k	0	1	2
y	1	2	3
$f(y)$	1	8	27

Полином Лагранжа второго порядка:

$$L(x) = 1 \cdot \frac{(x-2)(x-3)}{(1-2)(1-3)} + 8 \cdot \frac{(x-1)(x-3)}{(2-1)(2-3)} + 27 \cdot \frac{(x-1)(x-2)}{(3-1)(3-2)}$$
$$= 6\,x^2 - 11\,x + 6 \neq x^3$$

В результате аналитических вычислений получили неточную аппроксимацию (порядки $L(x)$ и $f(x)$ не совпали), причиной этого, в данном случае, является недостаточность исходного дискретного набора данных.

Пусть заданна таблично $f(x) = \tan(x)$:

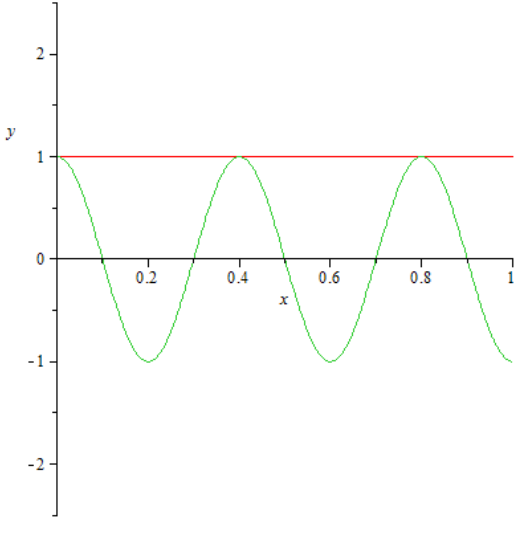
k	0	1	2	3	4
y	-1.5	-0.75	0	0.75	1.5
$f(y)$	-14.1	-0.93	0	0.93	14.1

Построим для нее полином Лагранжа 5-го порядка:

$$\begin{aligned} 243 \cdot L(x) = & f(x_0)\, x\, (2x-3)\, (4x-3)\, (4x+3) \\ & - 8 f(x_1)\, x\, (2x-3)\, (2x+3)\, (4x-3) \\ & + 3 f(x_2)\, (2x+3)\, (4x+3)\, (4x-3)\, (2x-3) \\ & - 8 f(x_3)\, x\, (2x-3)\, (2x+3)\, (4x+3) \\ & + f(x_4)\, x\, (2x+3)\, (4x-3)\, (4x+3) \\ = & \, 4.834848\, x^3 - 1.477474\, x \end{aligned}$$

Точная аппроксимация для не полиномиальной функции недостижима.

Графическая интерпретация полинома Лагранжа для $f(x) = \cos(2\pi x)$:



Интерполяция полиномом Ньютона

При интерполяции полиномом Ньютона функция $f(x)$ на интервале $[a, b]$ при $h = y_{k+1} - y_k = \text{const}$ аппроксимируется полиномом Ньютона порядка n:

$$P(x) = \sum_{k=0}^{n} f(y_k) \left(\prod_{i=0}^{k-1} (x - y_i) \right)$$

Интерполяция может осуществляться двумя подходами:

- первая интерполяционная формула Ньютона (вперед);
  (для интерполирования точек близких к a)
- вторая интерполяционная формула Ньютона (назад).
  (для интерполирования точек близких к b)

Первая интерполяционная формула Ньютона (вперед):

$$P(x) = f_0 + q\,\Delta f_0 + \frac{q\,(q-1)}{2!}\,\Delta^2 f_0 + \ \ldots\ + \frac{q\,(q-1)\cdots(q-n+1)}{n!}\,\Delta^n f_0\,,$$

где:

$$f_k = f(y_k)\,,$$
$$h = y_1 - y_0\,,$$
$$q = \frac{x - y_0}{h}\,,$$
$$\Delta f_k \leftarrow \text{конечная прямая разность первого порядка,}$$
$$\Delta^2 f_k \leftarrow \text{конечная прямая разность второго порядка,}$$
$$\vdots$$
$$\Delta^n f_k \leftarrow \text{конечная прямая разность порядка } n.$$

Первая интерполяционная формула Ньютона (вперед) $n = 1$:

$$P(x) = f_0 + q\,\Delta f_0$$

Первая интерполяционная формула Ньютона (вперед) $n = 2$:

$$P(x) = f_0 + q\,\Delta f_0 + \frac{q\,(q-1)}{2!}\,\Delta^2 f_0$$

Первая интерполяционная формула Ньютона (вперед) $n = 3$:

$$P(x) = f_0 + q\,\Delta f_0 + \frac{q\,(q-1)}{2!}\,\Delta^2 f_0 + \frac{q\,(q-1)(q-2)}{3!}\,\Delta^3 f_0$$

Точность интерполирования можно повысить за счет добавления новых точек и увеличения n, однако в отличии от использования полинома Лагранжа, весь $P(x)$ пересчитывать не потребуется

Вторая интерполяционная формула Ньютона (назад):

$$P(x) = f_n + q\,\nabla f_{n-1} + \frac{q\,(q+1)}{2!}\,\nabla^2 f_{n-2} + \ \ldots\ + \frac{q\,(q+1)\cdots(q+n-1)}{n!}\,\nabla^n f_0\,,$$

где:

$$f_k = f(y_k)\,,$$
$$h = y_n - y_{n-1}\,,$$
$$q = \frac{x - y_n}{h}\,,$$
$$\nabla f_k \leftarrow \text{конечная обратная разность первого порядка,}$$
$$\nabla^2 f_k \leftarrow \text{конечная обратная разность второго порядка,}$$
$$\vdots$$
$$\nabla^n f_k \leftarrow \text{конечная обратная разность порядка } n.$$

Конечные разности

Конечные разности перого порядка - это разности между значением функции в текущий и предыдущий моменты времени.

Обратная разность первого порядка:

$$\nabla f_k = f_k - f_{k-1}$$

Прямая разность первого порядка:

$$\Delta f_k = f_{k+1} - f_k$$

Обратная разность второго порядка:

$$\nabla^2 f_k = \nabla f_k - \nabla f_{k-1} = f_k - 2 f_{k-1} + f_{k-2}$$

Прямая разность второго порядка:

$$\Delta^2 f_k = \Delta f_{k+1} - \Delta f_k = f_{k+2} - 2 f_{k+1} + f_k$$

Обратная разность n-го порядка:

$$\nabla^n f_k = \nabla^{n-1} f_k - \nabla^{n-1} f_{k-1}$$

Прямая разность n-го порядка:

$$\Delta^n f_k = \Delta^{n-1} f_{k+1} - \Delta^{n-1} f_k$$

Вычисление обратной разности n-го порядка:

$$\nabla^n f_k = \sum_{v=0}^{n} (-1)^v \, C_n^v \, f_{k-v}, \qquad C_n^v = \frac{n!}{v!\,(n-v)!}$$

Вычисление прямой разности n-го порядка:

$$\Delta^n f_k = \sum_{v=0}^{n} (-1)^v \, C_n^v \, f_{k+n-v}$$

Существуют и центральные разности.

Пусть задано:

$$f(x) = x^4 + x^2 + 1.77$$
$$y_0 = 0.385$$
$$h = 0.2$$
$$k = \{0, 1, 2, 3\}$$
$$x = 0.885$$

Составим таблицу значений конечных разностей:

k	y_k	f_k	Δf_k	$\Delta^2 f_k$	$\Delta^3 f_k$
0	0.385	1.94	0.29	0.25	0.12
1	0.585	2.23	0.54	0.37	–
2	0.785	2.77	0.91	–	–
3	0.985	3.68	–	–	–

Используя первую интерполяционную формулу Ньютона (вперед):

$$q = \frac{0.885 - 0.385}{0.2} = 2.5$$

$$P(x = 0.885) = 1.94 + 2.5 \cdot 0.29 + \frac{2.5\,(2.5 - 1)}{2!} \cdot 0.25 + \frac{2.5\,(2.5 - 1)\,(2.5 - 1 - 1)}{3!} \cdot 0.12 = 3.1713$$

Используя вторую интерполяционную формулу Ньютона (назад):

$$q = \frac{0.885 - 0.985}{0.2} = -0.5$$

$$P(x = 0.885) = 3.68 + (-0.5) \cdot 0.91 + \frac{(-0.5)\,(-0.5 + 1)}{2!} \cdot 0.37 + \frac{(-0.5)\,(-0.5 + 1)\,(-0.5 + 1 + 1)}{3!} \cdot 0.12 = 3.1713$$

Теорема Шенберга-Уитни

Для $\forall f(x)$ и $\forall$ разбиения интервала $[a, b]$ существует единственный естественный сплайн $s(x)$, который на каждом участке $[y_{k-1}, y_k]$ является полином порядка $\leqslant 3$:

$$s_k(x) = a_k + b_k h_k + c_k h_k^2 + d_k h_k^3, \quad \text{где} \quad h_k = x - y_{k-1}$$

Интерполяция кубическими сплайнами

Заключается в вычислении набора коэффициентов $\{a_k, b_k, c_k, d_k\}$ сплайна $s_k(x)$ на интервалах $[y_{k-1}, y_k]$ для $k = 1, 2, \ldots, n$:

$$s_k(x) = a_k + b_k h_k + c_k h_k^2 + d_k h_k^3, \quad \text{где} \quad h_k = y_k - y_{k-1}$$

Вычисление значений функции $f(x)$ осуществляется подстановкой x в h_k и вычислением соответствующего $s_k(x)$.

Сплайны должны удовлетворять следующим условиям:

- условия непрерывности:

$$s_k^{(1)}(y_k) = s_{k+1}^{(1)}(y_k)$$
$$s_k^{(2)}(y_k) = s_{k+1}^{(2)}(y_k)$$

- условия Лагранжа:

$$s_k(y_{k-1}) = f(y_{k-1})$$
$$s_k(y_k) = f(y_k)$$

- условия естественности сплайна:

$$s_k^{(2)}(y_0) = 0$$
$$s_n^{(2)}(y_n) = 0$$

Задаются следующие начальные условия:

$$c_1 = c_{n+1} = K_1 = L_1 = 0$$

Итерационно вычисляются следующие значения:

$$\begin{aligned}
h_k &= y_k - y_{k-1} \\
h_{k-1} &= y_{k-1} - y_{k-2} \\
F_k &= 3\left(\frac{f_k - f_{k-1}}{h_k} - \frac{f_{k-1} - f_{k-2}}{h_{k-1}}\right) \\
V_k &= 2\left(h_k + h_{k-1}\right) \\
K_k &= \frac{F_k - h_{k-1} K_{k-1}}{V_k - h_{k-1} L_{k-1}}, \\
L_k &= \frac{h_k}{V_k - h_{k-1} L_{k-1}},
\end{aligned}$$

Начиная с n до 2, определяются значения c_k:

$$c_k = K_k - L_k\, c_{k+1}$$

По сути, вычисление c_k осуществляется методом прогонки - методом решения СЛАУ для трехдиагональных матриц.

Вычисляются значения остальных коэффициентов сплайнов:

$$d_k = \frac{c_{k+1} - c_k}{3\, h_k}$$

$$b_k = \frac{f_k - f_{k-1}}{h_k} - c_k\, h_k - d_k\, h_k^2$$

$$a_k = f_{k-1}$$

Интерполяция кубическими сплайнами реализуется в четыре цикла:

1. вычисление коэффициентов K_k и L_k;
   (остальные величины сохранять не требуется, кроме h_k и h_{k-1}, при этом сам цикл начинается с 2, т.к. K_1 и L_1 заданы как начальные условия)
2. вычисление коэффициентов c_k;
3. вычисление коэффициентов a_k, b_k и d_k;
   (цикл начинается с 1 элемента)
4. непосредственно сама интерполяция.
   (по аналогии с линейной интерполяцией при переходе от интервал к интервалу - от сплайна к сплайну)

Математически нумерация сплайнов, соответствующих каждому интервалу начинается с 1, а нумерация значений функции и аргумента с нуля.

Рациональная интерполяция

Рациональная интерполяция - это интерполирование функции, при котором $f(x)$ аппроксимируется не полиномом, а рациональной функцией, которая представляет собой отношение двух полиномов:

$$R(x) = \frac{a_0 + a_1 x + \ldots + a_p x^p}{b_0 + b_1 x + \ldots + b_q x^q},$$

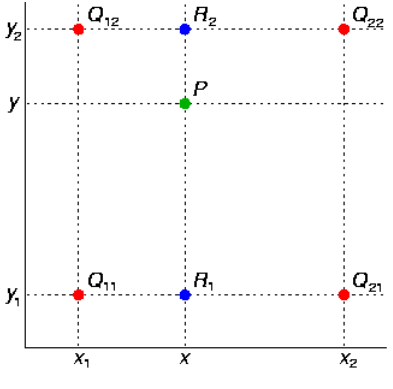
где $p + q + 1 = n$. В теории управления очень часто используется аппроксимация Паде $k -$ порядка для экспоненциальной функции (функция запаздывания).

Билинейная интерполяция

Билинейнея интерполяция - это расширение линейной интерполяции для функции $f(x, y)$ двух переменных, где по известным значениям функции в узловых точках $Q_{ij}(x_i, y_j)$ определяется значение функции в точке $P(x^*, y^*)$.

Координаты точек:

$$Q_{11} = (x_i, y_j)$$
$$Q_{12} = (x_i, y_{j+1})$$
$$Q_{21} = (x_{i+1}, y_j)$$
$$Q_{22} = (x_{i+1}, y_{j+1})$$
$$R_1 = (x^*, y_j)$$
$$R_2 = (x^*, y_{j+1})$$



Алгоритм билинейной интерполяции для одной точки заключается в последовательном линейном интерполировании по каждой координате.

Интерполируем во вспомогательных точках по координате x:

$$f(R_1) = \frac{x_{i+1} - x^*}{x_{i+1} - x_i} f(Q_{11}) + \frac{x^* - x_i}{x_{i+1} - x_i} f(Q_{21})$$

$$f(R_2) = \frac{x_{i+1} - x^*}{x_{i+1} - x_i} f(Q_{12}) + \frac{x^* - x_i}{x_{i+1} - x_i} f(Q_{22})$$

Интерполируем значение $f(P)$ по y:

$$f(P) = \frac{y_{j+1} - y^*}{y_{j+1} - y_j} f(R_1) + \frac{y^* - y_j}{y_{j+1} - y_j} f(R_2)$$

Для вычисления значений функции во всех точках (x^*, y^*) осуществляется проход по всей дискретной сетке по i и по j.

`interp1` - одномерная интерполяция;
`interp2` - двумерная интерполяция;
`spline` - интерполирование кубическими сплайнами.

Определенный интеграл по формуле Ньютона-Лейбница:

$$\int_a^b f(x)\,\mathrm{d}x = \Phi(b) - \Phi(a)$$

Сложность аналитического интегрирования заключается в том, что сложно (долго) искать первообразную, которая, ко всему прочему, может не существовать.

Идея численных методов интегрирования заключается в замене подынтегральной функции на более простую, интеграл от которой можно легко численно посчитать, т.е.:

$$\int_a^b f(x)\,\mathrm{d}x = \lim_{\Delta x \to 0} \sum_{i=0}^{N} f(x_i)\,\Delta x_i$$

Задача численного интегрирования

Заключается в замене подынтегральной функции более простой, с точки зрения вычислений, например, интерполяционным многочленом, в результате чего получается квадратурная формула вида:

$$\int_a^b f(x)\,\mathrm{d}x = \sum_{j=0}^{n} \omega_j f(x_j) + E(f),$$

где:

- n - количество узлов;
- x_j - узлы квадратурной формулы, при $j = 0, 1, 2, \ldots, n$;
- ω_j - весовые коэффициенты квадратурной формулы;
- $E(f)$ - погрешность вычисления.

- квадратурами интерполяционного типа:
  - квадратуры Ньютона-Котеса;
  - квадратуры Гаусса-Лежандра;
  - квадратуры Гаусса-Якоби;
  - квадратуры Чебышева-Гаусса;
  - квадратуры Гаусса-Лагерпа;
  - квадратуры Гаусса-Эрмита;
  - адаптивные квадратуры.
- стохастический метод (метод Монте-Карло).

Все квадратуры, за исключением Ньютона-Котеса, базируются на квадратурах Гаусса-Лежандра, как правило с заменой полинома Лежандра на другой полином

Квадратурные формулы интерполяционного типа

Сложная подынтегральная функция $f(x)$ заменяется (аппроксимируется) на более простую $g(x)$, представленную ввиде полинома, в результате чего приближенное значение интеграла рассчитывается как:

$$\int_a^b f(x)\,\mathrm{d}x \approx \int_a^b g(x)\,\mathrm{d}x$$

На практике участок $[a, b]$ разбивают на N интервалов $[x_k, x_{k+1}]$, каждый из которых аппроксимируется полиномиальной функцией $g(x)$, а значение интеграла в итоге вычисляется следующим образом:

$$\int_a^b f(x)\,\mathrm{d}x \approx \sum_{k=0}^{N-1} \int_{x_k}^{x_{k+1}} g(x)\,\mathrm{d}x$$

Квадратурные формулы Ньютона-Котеса

Квадратурные формулы Ньютона-Котеса это формулы вида:

$$\int_a^b f(x)\,\mathrm{d}x \approx \sum_{j=0}^{n} \omega_j f(x_j)\,,$$

которые строятся со следующими предположениями:

- узлы квадратурных формул x_j фиксированы (в частном случае равноудалены друг от друга);
- вся функция либо интервал $[x_k, x_{k+1}]$ аппроксимируется (глобально или локально) кусочно-линейной интерполяцией;
- в качестве полинома используется полином Лагранжа.

Случай глобальной аппроксимации

Пусть имеется фиксированный набор n узлов $x_j \in [a, b]$ при $j = 0, 1, 2, \ldots, n$ (как правило, равноотстоящих), известны значения подынтегральной функции $f_j = f(x_j)$. Аппроксимируем функцию $f(x)$ полиномом Лагранжа порядка n на всем интервале $[a, b]$:

$$L_n(x) = \sum_{j=0}^{n} f(x_j) \prod_{i=1,\, i \neq j}^{n} \frac{x - x_i}{x_j - x_i}$$

Соответственно:

$$\int_a^b f(x)\,\mathrm{d}x \approx \int_a^b g(x)\,\mathrm{d}x \approx \int_a^b L_n(x)\,\mathrm{d}x$$

В итоге получается обобщенная квадратурная формула

Случай локальной аппроксимации

В этом случае $f(x)$ аппроксимируется на интервале $[x_k, x_{k+1}]$ аналогично полиномом Лагранжа порядка m:

$$L_m(x) = \sum_{j=0}^{m} f(x_j) \prod_{i=1,\, i\neq j}^{m} \frac{x - x_i}{x_j - x_i}$$

Соответственно:

$$\int_a^b f(x)\,\mathrm{d}x \approx \sum_{k=0}^{N-1} \int_{x_k}^{x_{k+1}} g(x)\,\mathrm{d}x \approx \sum_{k=0}^{N-1} \int_{x_k}^{x_{k+1}} L_m(x)\,\mathrm{d}x$$

В итоге получается составная квадратурная формула

Составная квадратурная формула в общем виде:

$$\int_a^b f(x)\,\mathrm{d}x \approx \sum_{k=0}^{N-1} \int_{x_k}^{x_{k+1}} L_m(x)\,\mathrm{d}x$$

$L_m(x_j)$ должен проходить через точки $f(x_j)$ (как полином Лагранжа)
Интегрируя $L_m(x_j)$, можем перейти к составным квадратурным формулам для заданного порядка m

Порядок m метода:

- $m = 0$ - метод прямоугольников;
- $m = 1$ - метод трапеций;
- $m = 2$ - метод парабол (метод Симпсона);
- ... - и т.д.

Составная квадратурная формула:

$$\int_a^b f(x)\,\mathrm{d}x \approx \frac{1}{C_m} \sum_{k=0}^{N-1} h_k \left(\sum_{i=0}^{m} C_{im} f(x_i) \right),$$

где:

- $h_k = x_{k+1} - x_k$ - шаг;
- $x_i = x_k + i\,h$ - значение узлов внутри интервала;
- $C_m = \sum_{i=0}^{m} C_{im}$ - сумма коэффициентов Ньютона-Котеса.

В таблице ниже приведены значения коэффициентов для ОБОБЩЕННЫХ квадратурных формул

m	C_m	C_{0m}	C_{1m}	C_{2m}	C_{3m}	C_{4m}	C_{5m}	Название метода
0	1	1	–	–	–	–	–	Прямоугольники
1	2	1	1	–	–	–	–	Трапеции
2	6	1	4	1	–	–	–	Формула Симпсона
3	8	1	3	3	1	–	–	3/8 Симпсона
4	90	7	32	12	32	7	–	Формула Милна
5	188	19	75	50	50	75	19	

В методах прямоугольников интервал $[a, b]$ разбивается на N интервалов.

Метод левых прямоугольников:

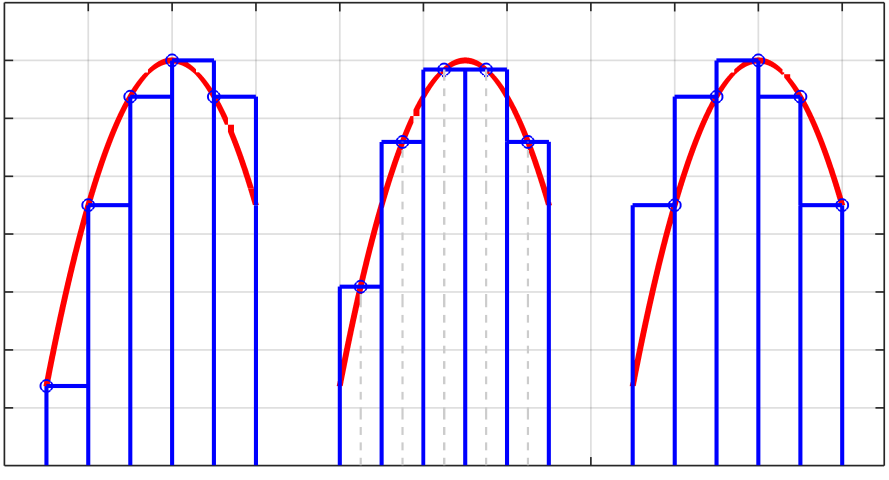
$$\int_a^b f(x)\,\mathrm{d}x = \sum_{i=0}^{N-1} f(x_i)\,(x_{i+1} - x_i) + E(f)$$

Метод правых прямоугольников:

$$\int_a^b f(x)\,\mathrm{d}x = \sum_{i=1}^{N} f(x_i)\,(x_i - x_{i-1}) + E(f)$$

Метод средних прямоугольников:

$$\int_a^b f(x)\,\mathrm{d}x = \sum_{i=1}^{N} f\left(\frac{x_{i-1} + x_i}{2}\right)\left(x_i - x_{i-1}\right) + E(f)$$



слева метод левых прямоугольников;
в центре метод средних прямоугольников;
справа метод правых прямоугольников.

В методе трапеций интервал $[a, b]$ также разбивается на N интервалов.

Составная квадратурная формула:

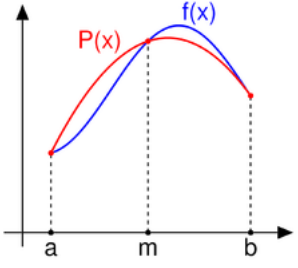
$$\int_a^b f(x)\,\mathrm{d}x = \sum_{i=0}^{N-1} \frac{f(x_i) + f(x_{i+1})}{2}(x_{i+1} - x_i) + E(f)$$

$$= \frac{f(a)}{2}(x_1 - a) + \frac{f(b)}{2}(b - x_{N-1}) + \sum_{i=1}^{N-1} f(x_i)\frac{x_{i+1} - x_{i-1}}{2} + E(f)$$

В методе Симпсона мы разбиваем интервал $[a, b]$ на N интервалов, а затем сам метод доразбивает каждый подинтервал на два, делая общее количество интервалов равное $2N$.

Составная квадратурная формула (при $h = \text{const}$):

$$\int_a^b f(x)\,\mathrm{d}x = \frac{h}{3}\left(f(x_0) + f(x_{2N}) + 2\sum_{i=1}^{N-1} f(x_{2i}) + 4\sum_{i=1}^{N} f(x_{2i-1}) \right)$$



Алгебраический порядок точности

Алгебраический порядок точности численного метода - это наибольшая степень полинома, для которой численный метод даёт точное решение.

Альтернативное определение: численный метод имеет порядок точности d, если его остаток $E(f)$ равен нулю для любого полинома степени d, но не равен нулю для полинома степени $d+1$.

По n точкам можно получить порядок точности $2n-1$
(речь о $m-1$ точках каждого отрезка $[x_j, x_{j+1}]$)

Для равноудаленных узлов максимальный порядок точности равен двум

Методы повышения точности интегрирования:

- усложнение глобальной аппроксимации;
  (аппроксимация всей $f(x)$)
- усложнение локальной аппроксимации;
  (аппроксимация $f(x)$ на интервале $[x_j, x_{j+1}]$)
- увеличение количества разбиений (интервалов) $[x_j, x_{j+1}]$;
  (квадратурные формулы Ньютона-Котеса)
- выбор узлов x_j.
  (квадратурные формулы Гаусса)

Алгебраический порядок точности некоторых методов:

- 0 - метод прямоугольников (без средних);
- 1 - метод трапеций;
- 3 - метод парабол (метод Симпсона);
- 9 - метод Гаусса по пяти точкам.

Формула Рунге:

$$\Delta_{2N} \approx \Theta \left| I_{2N} - I_N \right|$$

где:

I_N	-	значение интеграла, вычисленное для N интервалов;
I_{2N}	-	значение интеграла, вычисленное для $2N$ интервалов;
$\Theta = 1/3$	-	для методов прямоугольников и трапеций;
$\Theta = 1/15$	-	для метода Симпсона;

Процесс интегрирования, где на каждой итерации увеличивается количество разбиений в два раза, заканчивается при выполнении условия:

$$\Delta_{2N} < \varepsilon$$

Квадратурные формулы Гаусса

Квадратурные формулы Гаусса - это формулы вида:

$$\int_{-1}^{1} f(x)\,\mathrm{d}x = \int_{-1}^{1} p(x)\,g(x)\,\mathrm{d}x \approx \sum_{j=1}^{n} \omega_j\, f(x_j)\,,$$

которые строятся со следующими предположениями:

- узлы $x_j \in [-1, 1]$ при $j = 1, 2, \ldots, n$ не фиксированы;
- $p(x)$ - весовая функция, определяющая конкретную квадратуру.
- переход к интегрированию интервала $[a, b]$ или интервала между полученными ранее разбиениями зависит от конкретного вида квадратуры и вариантов интервалов интегрирования.

$f(x)$ ДОЛЖНА БЫТЬ ЗАДАНА В ЯВНОМ (АНАЛИТИЧЕСКОМ) ВИДЕ

Вид весовой функции $p(x)$ определяет конкретную квадратуру:

Интервал	$p(x)$	Квадратура
$[-1, 1]$	1	Гаусса-Лежандра
$(-1, 1)$	$(1-x)^{\alpha}(1-x)^{\beta}, \quad \alpha, \beta > -1$	Гаусса-Якоби
$(-1, 1)$	$\dfrac{1}{\sqrt{1-x^2}}$	Чебышева-Гаусса I типа
$[-1, 1]$	$\sqrt{1-x^2}$	Чебышева-Гаусса II типа
$[0, \infty)$	e^{-x}	Гаусса-Лагерра
$[0, \infty)$	$x^{\alpha}\mathrm{e}^{-x}, \quad \alpha > -1$	Гаусса-Лагерра
$(-\infty, \infty)$	e^{-x^2}	Гаусса-Эрмита

Квадратурные формулы Гаусса-Лежандра

На интервале $[-1, 1]$ при $p(x) = 1$ квадратурная формула имеет вид:

$$\int_a^b f(x)\,\mathrm{d}x \approx \frac{b-a}{2} \sum_{j=1}^{n} \omega_j f\left(\frac{b-a}{2} x_j + \frac{a+b}{2}\right),$$

где весовые коэффициенты ω_j определяются как:

$$\omega_j = \frac{2}{\left(1 - x_j^2\right)\ \left[P_n'(x_j)\right]^2},$$

где P_n' - первая производная полинома Лежандра:

$$P_n(x) = \frac{1}{2^n\, n!}\ \frac{\mathrm{d}^n}{\mathrm{d}x^n}\ \left(x^2 - 1\right)^n$$

Случай глобальной аппроксимации:

$$\int_a^b f(x)\,\mathrm{d}x \approx \frac{b-a}{2}\sum_{j=1}^{n}\omega_j f\left(\frac{b-a}{2}x_j+\frac{a+b}{2}\right),$$

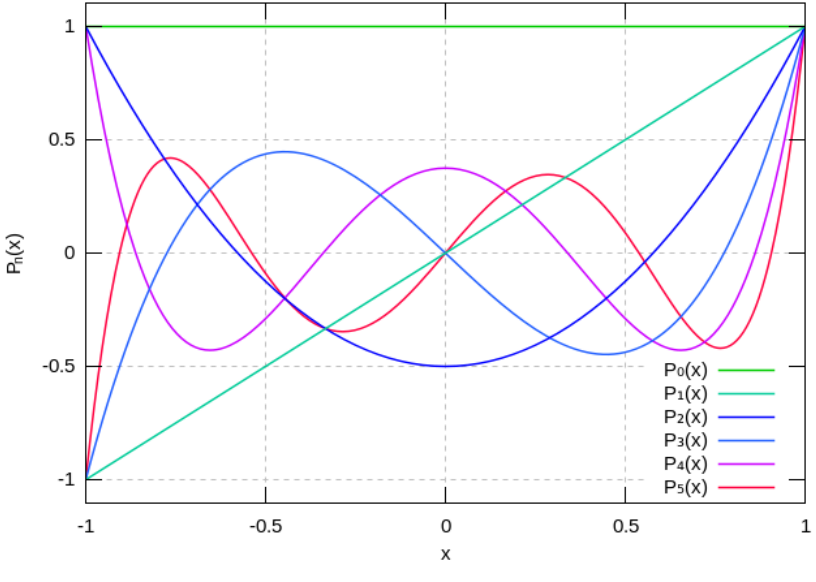
n точек распределены на весь интервал $[a,b]$

Случай локальной аппроксимации:

$$\int_a^b f(x)\,\mathrm{d}x \approx \sum_{k=0}^{N-1}\int_{x_k}^{x_{k+1}} f(x)\,\mathrm{d}x$$

$$\approx \sum_{k=0}^{N-1}\frac{x_{k+1}-x_k}{2}\sum_{j=1}^{n}\omega_j f\left(\frac{x_{k+1}-x_k}{2}x_j+\frac{x_k+x_{k+1}}{2}\right),$$

n точек распределены в каждом интервале $[x_k,x_{k+1}]$


1
0.5
0
-0.5
-1
Pn(x)
-1
-0.5
0
0.5
1
x
P0(x)
P1(x)
P2(x)
P3(x)
P4(x)
P5(x)

Количество точек, n	узлы x_j	веса ω_j	Порядок точности
1	0	2	1
2	$\pm$0.57735	1	2
3	0	0.888889	5
	$\pm$0.774597	0.555556	
4	$\pm$0.339981	0.652145	7
	$\pm$0.861136	0.347855	
5	0	0.568889	9
	$\pm$0.538469	0.478629	
	$\pm$0.90618	0.236927	

Квадратурные формулы Гаусса-Кронрода

К n-точечной квадратуре Гаусса добавляют $n+1$ дополнительных узлов. Погрешности определяется как разность для вычисленного значения интеграла при n и $2n+1$ узлах. Например, G7-K15:

узлы x_j	веса ω_j	точка
$\pm$0.99145 53711 20813	0.02293 53220 10529	Кронрод
$\pm$0.94910 79123 42759	0.06309 20926 29979	Гаусс
$\pm$0.86486 44233 59769	0.10479 00103 22250	Кронрод
$\pm$0.74153 11855 99394	0.14065 32597 15525	Гаусс
$\pm$0.58608 72354 67691	0.16900 47266 39267	Кронрод
$\pm$0.40584 51513 77397	0.19035 05780 64785	Гаусс
$\pm$0.20778 49550 07898	0.20443 29400 75298	Кронрод
0	0.20948 21410 84728	Гаусс

Адаптивные квадратуры

Рассмотренные ранее квадратурные формулы применяются к каждому из N равных интервалов, полученному в результате разбиения $[a, b]$:

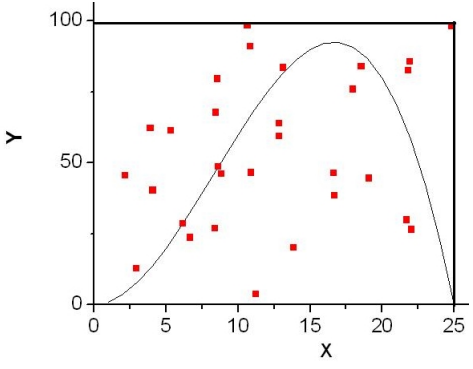
$$\int_a^b f(x)\,\mathrm{d}x \approx \sum_{k=0}^{N-1} \int_{x_k}^{x_{k+1}} g(x)\,\mathrm{d}x$$

Адаптивные квадратуры используют некоторый алгоритм автоматического распределения узлов интегрирования, например, постепенное деление интервала пополам, до тех пор, пока не выполнится условие:

$$\sum_{k=0}^{N-1} \left| \int_{x_k}^{x_{k+1}} f(x)\,\mathrm{d}x \Big|_{h} - \int_{x_k}^{x_{k+1}} f(x)\,\mathrm{d}x \Big|_{0.5\,h} \right| < \varepsilon$$

Стохастический метод интегрирования

Стохастический метод интегрирования (метод Монте-Карло) - это метод в котором фигуры определяется с использованием статистических испытаний.



Идея метода:

- выбираем n-мерный параллелепипед с площадью S_{par};
- генерируем N случайных точек с координатами $x_1, \ldots, x_n$;
- считаем количество K точек, попавших в область графика;
- определяем значение интеграла:

$$I = S_{\text{par}} \frac{K}{N}$$

Для достаточной точности метод требует большого объема вычислений, но в некоторых случаях является единственным применимым подходом.

`quadgk` - метод интегрирования Гаусса-Кронрода;
`trapz` - интегрирование методом трапеций;
`integral` - интегрирование адаптивными квадратурами;
`sum` - сумма элементов вектора.

Динамические системы

Объект или процесс, описываемый совокупностью некоторых величин (переменных), изменяющихся в процессе его работы по времени.

Непрерывные системы

Непрерывные системы представляют собой объекты или процессы, происходящие в окружающем нас мире. Математическое описание непрерывных систем строится на различных видах дифференциальных уравнений.

Дискретные системы

Системы, в которых изменение состояния объекта или процесса нельзя назвать непрерывным (как правило, обязательно присутствует дискретность по времени), являются дискретными. Математическое описание таких систем основывается на разностных уравнениях.

Виды динамических систем:

- линейные и нелинейные;
- стационарные и нестационарные;
- непрерывные, дискретные и смешанные;
- детерминистические и стохастические;
- системы со сосредоточенными и распределенными параметрами;
- адаптивные системы;
- автономные системы.

Математический аппарат описания моделей:

- дифференциальные уравнения; *(непрерывные системы)*
- разностные уравнения; *(дискретные системы)*
- переменные вход–выход; *(только линейные системы)*
- пространство состояний. *(только линейные системы)*

Система

Совокупность объектов, функционирующих друг с другом для достижения определенной (заданной) цели. Систему с множественными связями называют сложной.

Состояние системы

Совокупность переменных, необходимых для ее описания в конкретный момент времени. Под переменными понимаются некоторые физические величины, например, расстояние, скорость и т.п.

Модель

Идеализированное представление, обладающее определенной (достаточной для решения задачи моделирования) степенью адекватности (подобия) исследуемой системы (объекта, процесса).

Адекватность модели

Основополагающее свойство для моделирования. Адекватность модели это степень соответствия модели (описания системы) поведению (изменению, динамики) исследуемой системе (объекту, процессу).

Оценка адекватности, как правило, проводится по косвенным показателям, например, сравнением выхода моделируемой системы с выходом системы, полученным в ходе эксперимента, при условии одинаковости (на сколько это возможно) моделируемых условий проведения экспериментальным.

Ошибка моделирования

С практической точки зрения - количественная оценка отклонения результатов моделирования от полученных результатов в ходе эксперимента.

Математическая модель

Описание исследуемой системы (объекта, процесса) одним из существующих математических аппаратов, например:

- дифференциальные уравнения;
- интегро-дифференциальные уравнения;
- разностные уравнения;
- алгебраические уравнения;
- вероятностные модели;
- нейросетевые модели;
- и др.

Дельта-функция Дирака

Обобщенная функция для описания точечного воздействия:

$$\delta(t) = \begin{cases} +\infty & t = 0 \\ 0 & t \neq 0 \end{cases}$$

Дельта-функция Дирака обладает важными свойствами:

$$\int_{-\infty}^{+\infty} \delta(t)\,\mathrm{d}t = 1$$

$$\int_{-\infty}^{+\infty} f(t)\,\delta(t - t_0)\,\mathrm{d}t = f(t_0)$$

Функция Хевисайда

Кусочно-непрерывная функция, которая, как правило, считается “ступенчатым” воздейтсвием. Для непрерывного случая имеет два математических определения:

$$H(t) = \begin{cases} 0, & t < 0 \\ 1/2, & t = 0 \\ 1, & t > 0 \end{cases} \quad \text{или} \quad H(t) = \begin{cases} 0, & t < 0 \\ 1, & t \geqslant 0 \end{cases}$$

Функция Хевисайда - первообразная дельта-функции Дирака:

$$H(t) = \int\limits_{-\infty}^{t} \delta(\tau)\,\mathrm{d}\tau$$

Интегральное преобразование

Интегральное преобразование - это преобразование (перевод) функции из одной области применения (например, временной) в другую область применения (например, частотную), осуществляемое интегрированием заданной функции.

Как правило, в теории управления применяют преобразование из временной в частотную область, которое либо позволяет более простым образом получать решения различных видов дифференциальных уравнений, либо дает возможность исследовать сигнал или систему в частотной области.

Виды интегральных преобразований:

- преобразование Лапласа;
- преобразование Фурье (непрерывное);
- и др.

Прямое преобразование:

$$f(\omega) = \mathcal{T}\left\{ f(t) \right\} = \int_{t_1}^{t_2} K(t,\omega)\, f(t)\ \mathrm{d}t$$

Обратное преобразование:

$$f(t) = \mathcal{T}^{-1}\left\{ f(\omega) \right\} = \int_{\omega_1}^{\omega_2} K^{-1}(t,\omega)\, f(\omega)\ \mathrm{d}\omega$$

где:

- $f(t)$ - оригинал;
- $f(\omega)$ - изображение;
- $\mathcal{T}$ - обозначение преобразования;
- $K(t,\omega)$ - ядро интегрального преобразования.

Интегральное преобразование Лапласа

Интегральное преобразование Лапласа - это интегральное преобразование функции, определенной во временной области $f(t)$, в функцию, определенную в частотной области $f(s)$ (при $s = \jmath\omega$). Основное назначение - аналитическое решение дифференциальных уравнений.

Прямое преобразование:

$$f(s) = \mathcal{L}\left\{ f(t) \right\} = \int_0^\infty \mathrm{e}^{-st} f(t)\, \mathrm{d}t$$

Обратное преобразование:

$$f(t) = \mathcal{L}^{-1}\left\{ f(s) \right\} = \frac{1}{2\pi\jmath} \int_{-\infty}^{\infty} \mathrm{e}^{st} f(s)\, \mathrm{d}s$$

- линейность:

$$\mathcal{L}\left\{ a\,f(t) + b\,g(t) \right\} = a\,\mathcal{L}\left\{ f(t) \right\} + b\,\mathcal{L}\left\{ g(t) \right\}$$

- подобие:

$$\mathcal{L}\left\{ f(at) \right\} = \frac{1}{a} f\left(\frac{s}{a} \right)$$

- абсолютная сходимость:

  Если интеграл Лапласа абсолютно сходится при $\sigma = \sigma_0$, то

$$\lim_{b\to\infty} \int_0^b \left| f(t) \right| \mathrm{e}^{-\sigma_0 t}\,\mathrm{d}t = \int_0^\infty \left| f(t) \right| \mathrm{e}^{-\sigma_0 t}\,\mathrm{d}t$$

  существует и сходится равномерно и абсолютно для $\sigma \geqslant \sigma_0$.

- дифференцирование оригинала:
  Если $f(t), f^{(1)}(t), \ldots, f^{(n)}(t)$ являются оригиналами, то:

$$\mathcal{L}\left\{ f^{(0)}(t) \right\} = f(s)$$

$$\mathcal{L}\left\{ f^{(1)}(t) \right\} = s f(s) - f(0)$$

$$\mathcal{L}\left\{ f^{(2)}(t) \right\} = s^2 f(s) - s f(0) - f^{(1)}(0)$$

$$\vdots$$

$$\mathcal{L}\left\{ f^{(n)}(t) \right\} = s^n f(s) - \sum_{i=1}^{n-1} s^i f(0) - \sum_{i=1}^{n-1} f^{(i)}(0)$$

- интегрирование оригинала:

  Если $f(t)$ - оригинал, а $f(s)$ его изображением по Лапласу, то

$$\mathcal{L}\left\{\int_0^t f(\tau)\,\mathrm{d}\tau\right\} = \frac{f(s)}{s}$$

- запаздывание оригинала:

  Если $f(t)$ - оригинал, а $f(s)$ его изображением по Лапласу, то

$$\mathcal{L}\left\{f(t-\tau)\right\} = f(s)\,\mathrm{e}^{-s\tau}$$

- дифференцирование изображения:

  Если $f(t)$ - оригинал, а $f(s)$ его изображением по Лапласу, то

$$\mathcal{L}\left\{(-1)^n t^n f(t)\right\} = f^{(n)}(s)$$

- интегрирование изображения:

  Если $f(t)/t$ - оригинал, то

$$\mathcal{L}\left\{\frac{f(t)}{t}\right\} = \int_s^\infty f(z)\,\mathrm{d}z$$

- теорема о начальном и конечном значениях:

  дает возможность определить значение оригинала функции при $t \to \infty$ и при $t = 0$ по известному изображению.

  $$f(0) = \lim_{s \to \infty} s f(s)$$

  Если существует конечный предел

  $$\lim_{t \to \infty} f(t) = f(\infty)$$

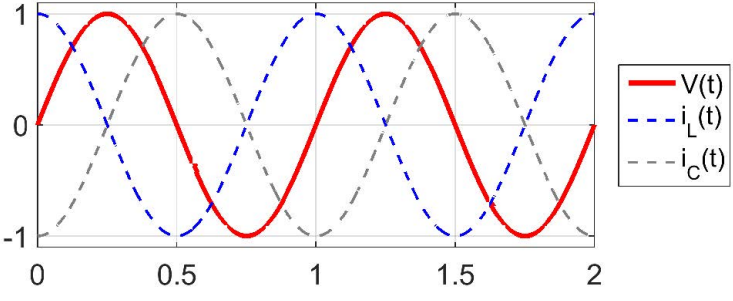
  тогда

  $$f(\infty) = \lim_{s \to 0} s f(s)$$

$$V(t) = \sin(\omega t)$$

$$i_L(t) = \frac{1}{L} \int_0^t V(t)dt$$

$$i_C(t) = C \frac{\mathrm{d}V(t)}{\mathrm{d}t}$$



Найдем токи:

$$i_L(t) = \frac{1}{L} \int_0^t \sin(\omega t)\,\mathrm{d}t = \frac{1}{\omega L}\left(-\cos(\omega t)\right) = \frac{1}{\omega L}\cos(\omega t - \pi)$$

$$= \frac{1}{\omega L}\sin\left(\omega t - \pi + \frac{\pi}{2}\right) = \frac{1}{\omega L}\sin\left(\omega t - \frac{\pi}{2}\right)$$

$$i_C(t) = C\frac{\mathrm{d}\sin(\omega t)}{\mathrm{d}t} = \omega C\cos(\omega t) = \omega C\sin\left(\omega t + \frac{\pi}{2}\right)$$

$f(t)$	$f(s)$	$f(t)$	$f(s)$
$\delta(t)$	1	$\delta(t-\tau)$	$\mathrm{e}^{-s\tau}$
$H(t)$	$\dfrac{1}{s}$	$H(t-\tau)$	$\dfrac{\mathrm{e}^{-s\tau}}{s}$
$\mathrm{e}^{-a\tau}\,H(t)$	$\dfrac{1}{s+a}$	$(1-\mathrm{e}^{-a\tau})\,H(t)$	$\dfrac{a}{s(s+a)}$
$t\,H(t)$	$\dfrac{1}{s^2}$	$t\,\mathrm{e}^{-at}\,H(t)$	$\dfrac{1}{(s+a)^2}$
$\dfrac{t^n}{n!}\,H(t)$	$\dfrac{1}{s^{n+1}}$	$\dfrac{t^n}{n!}\mathrm{e}^{-at}\,H(t)$	$\dfrac{1}{(s+a)^{n+1}}$

$f(t)$	$f(s)$	$f(t)$	$f(s)$
$\sin(\omega t)\,H(t)$	$\dfrac{\omega}{s^2+\omega^2}$	$\sin(\omega t)\,\mathrm{e}^{-at}\,H(t)$	$\dfrac{\omega}{(s+a)^2+\omega^2}$
$\cos(\omega t)\,H(t)$	$\dfrac{s}{s^2+\omega^2}$	$\cos(\omega t)\,\mathrm{e}^{-at}\,H(t)$	$\dfrac{s+a}{(s+a)^2+\omega^2}$
$\mathrm{sh}(at)\,H(t)$	$\dfrac{a}{s^2-a^2}$		
$\mathrm{ch}(at)\,H(t)$	$\dfrac{s}{s^2-a^2}$		

для дифференциального уравнения:

$$x^{(2)}(t) + a_1 x^{(1)}(t) + a_2 x(t) = f(t)$$

удовлетворяющего начальным условиям:

$$x(0) = x_0\,, \quad x^{(1)}(0) = x_1$$

запишем производные, используя преобразование Лапласа:

$$\mathcal{L}\left\{ x^{(0)}(t) \right\} = x(s)$$

$$\mathcal{L}\left\{ x^{(1)}(t) \right\} = s\,x(s) - x(0) = s\,x(s) - x_0$$

$$\mathcal{L}\left\{ x^{(2)}(t) \right\} = s^2\,x(s) - s\,x(0) - x^{(1)}(0) = s^2\,x(s) - s\,x_0 - x_1$$

перепишем исходное дифференциальное уравнение в виде:

$$(s^2 + a_1\,s + a_2)x(s) = f(s) + x_0(s + a) + x_1$$

решая алгебраическое уравнение относительно $x(s)$:

$$(s^2 + a_1 s + a_2)\, x(s) = f(s) + x_0 (s + a) + x_1$$

получим:

$$x(s) = \frac{f(s) + x_0 (s + a) + x_1}{s^2 + a_1 s + a_2}$$

раскладывая $x(s)$ на элементарные дроби $g_i(s)$ и применяя обратное преобразование Лапласа, находим решение относительно $x(t)$:

$$x(t) = \mathcal{L}^{-1}\left\{ x(s) \right\} = \mathcal{L}^{-1}\left\{ \sum_i g_i(s) \right\} = \sum_i \mathcal{L}^{-1}\left\{ g_i(s) \right\}$$

$$= \sum_i g_i(t)$$

Разложение на элементарные дроби позволяет использовать таблицы преобразований Лапласа

для дифференциального уравнения:

$$x^{(2)}(t) + 2\,x^{(1)}(t) + 2\,x(t) = t\,\mathrm{e}^{-t}$$

удовлетворяющего начальным условиям:

$$x(0) = 0\,, \quad x^{(1)}(0) = 0$$

запишем производные и $f(t)$, используя преобразование Лапласа:

$$\mathcal{L}\left\{ x^{(0)}(t) \right\} = x(s)$$

$$\mathcal{L}\left\{ x^{(1)}(t) \right\} = s\,x(s) - x(0) = s\,x(s)$$

$$\mathcal{L}\left\{ x^{(2)}(t) \right\} = s^2\,x(s) - s\,x(0) - x^{(1)}(0) = s^2\,x(s)$$

$$\mathcal{L}\left\{ t\,\mathrm{e}^{-t} \right\} = \frac{1}{(s+1)^2}$$

получим алгебраическое уравнение:

$$x(s) = \frac{1}{(s+1)^2\,(s^2+2\,s+2)}$$

разложив на элементарные дроби получим:

$$x(s) = \frac{1}{(s+1)^2\,(s^2+2\,s+2)} = \frac{1}{(s+1)^2} - \frac{1}{(s+1)^2+1}$$

используя соотношения из таблицы преобразования Лапласа:

$$\mathcal{L}^{-1}\left\{\frac{1}{(s+1)^2}\right\} = t\,\mathrm{e}^{-t}$$

$$\mathcal{L}^{-1}\left\{\frac{1}{(s+1)^2+1}\right\} = \mathrm{e}^{-t}\,\sin(t)$$

запишем решение $x(t)$:

$$x(t) = t\,\mathrm{e}^{-t} - \mathrm{e}^{-t}\,\sin(t) = \mathrm{e}^{-t}\,(t - \sin(t))$$

Моделирование

Метод научного познания, используемый для исследования сложных систем (объектов, процессов). Моделирование заключается в замене реально существующей или проектируемой сложной системы моделью, свойства, характеристики и поведение которой исследуется. Получаемые результаты принимаются характерными для исходной сложной системы.

Задача моделирования

Задача моделирования заключается в создании (построении через процедуру формализации) модели сложной системы с последующим построением и проведением эксперимента над моделью и анализом результатов.

- проектирование сложных систем;
  (в качестве предварительного анализа работы будущей системы)
- в процессе эксплуатации сложных систем:
  - для контроля и тестирования;
    (работа системы сравнивается с работой модели)
  - непосредственное участие;
    (например, тренажеры подготовки летного состава)
- тестирование / экспериментирование;
  (проведение научных экспериментов)
- поиск причины неисправности, поломки.
  (моделируется критическая ситуация, вызвавшая поломку)

Математическое моделирование

Исследование системы (объекта, процесса) с предварительно полученной моделью, выраженной в некоторой математической форме записи.

Задача формализации

Задача формализации - получение математического описания, т.е. модели, (с помощью различных видов преобразований) исследуемой системы (объекта, процесса).

Виды математического моделирования:

- аналитическое;
- имитационное;
- комбинированное.

Аналитическое моделирование

Исследование системы, заданной (описанной) с использованием: дифференциальных уравнений, интегро-дифференциальных уравнений, разностных уравнений, алгебраических уравнений и др. Существует три метода аналитического моделирования:

- аналитический;
  (получение решения уравнений, описывающих систему в явном, т.е. символьном виде)
- численный;
  (получение решения уравнений, описывающих систему в виде набора численных значений)
- качественный.
  (оценка некоторых характеристик или свойств модели, без получения решения, например, оценка устойчивости динамических систем алгебраическими критериями)

Динамика непрерывной системы (электрика):

$$LC\,\frac{\mathrm{d}^2 u_C(t)}{\mathrm{d}t^2} + RC\,\frac{\mathrm{d}u_C(t)}{\mathrm{d}t} + u_C(t) = -u(t)$$

Динамика непрерывной системы (механика):

$$M\,\frac{\mathrm{d}^2 x(t)}{\mathrm{d}t^2} + B\,\frac{\mathrm{d}x(t)}{\mathrm{d}t} + K\,x(t) = f(t)$$

Моделирование заключается в решении задачи Коши, т.е. в данных конкретных примерах в получении в аналитическом виде:

$$u_c(t) = \ldots \quad \text{или} \quad x(t) = \ldots$$

с последующей записью полученных функций в виде операторов языка программирования и в подстановке значений времени t .

Поиск решения аналитическими способами затруднен
На практике используются численные методы решения ДУ

Динамика непрерывной системы (механика):

$$W(s) = \frac{x(s)}{u(s)} = \frac{1}{M s^2 + B s + K}$$

Динамика непрерывной системы (электрика):

$$W(s) = \frac{u_c(s)}{u(s)} = \frac{-1}{LC s^2 + RC s + 1}$$

Моделирование заключается в аналитическом решении алгебраических уравнений относительно $x(s)$ или $U_c(s)$, например:

$$x(s) = W(s)\, u(s)$$

после чего с использованием обратного преобразования Лапласа получается решение относительно $x(t)$ или $U_c(t)$. Последующий подход моделирования аналогичен случаю с дифференциальными уравнениями.

Описание системы в переменных состояния имеет вид:

$$\begin{cases} \dot{\mathbf{x}}(t) = \mathbf{A}(t)\,\mathbf{x}(t) + \mathbf{B}(t)\,\mathbf{u}(t) + \mathbf{G}\,\omega(t) \\ \mathbf{y}(t) = \mathbf{C}(t)\,\mathbf{x}(t) + \mathbf{D}(t)\,\mathbf{u}(t) + \mathbf{H}\,\upsilon(t) \end{cases}$$

Моделирование подобной системы заключается в решении классической задачи Коши для матричного дифференциального уравнения. Решение получается в следующем виде:

$$\mathbf{x}(t) = \mathrm{e}^{\mathbf{A}\,t}\,\mathbf{x}_0 = \left(\mathbf{I} + \mathbf{A}\,t + \frac{\mathbf{A}^2\,t^2}{2!} + \ldots\right)\mathbf{x}_0 = \Phi(t)\,\mathbf{x}_0$$

$$\mathbf{y}(t) = \ldots \longleftarrow \text{получаем из уравления выхода}$$

$\Phi(t)$ - фундаментальная матрица решений, вычисление которой весьма трудоемко для больших систем.

`tf` - задание модели в переменных вход-выход;
`ss` - задание модели в переменных состояния;
`ss2tf` - перевод описание в переменных вход выход;
`tfdata` - коэффициенты числителя и знаменателя $W(s)$;
`series` - последовательное соединение;
`parallel` - параллельное соединение;
`feedback` - обратная связь.

В основе анализа динамических систем лежит построение основных графиков, которые наглядно отображают основные характеристики системы.

Временные характеристики:

- построение графика переходного процесса;
- построение графика импульсной переходной характеристики;

Частотные характеристики:

- вычисление частотной характеристики;
- построение диаграммы Боде; *(ЛАФЧХ)*
- построение годографа Найквиста; *(АФЧХ)*

Нули и полюса системы:

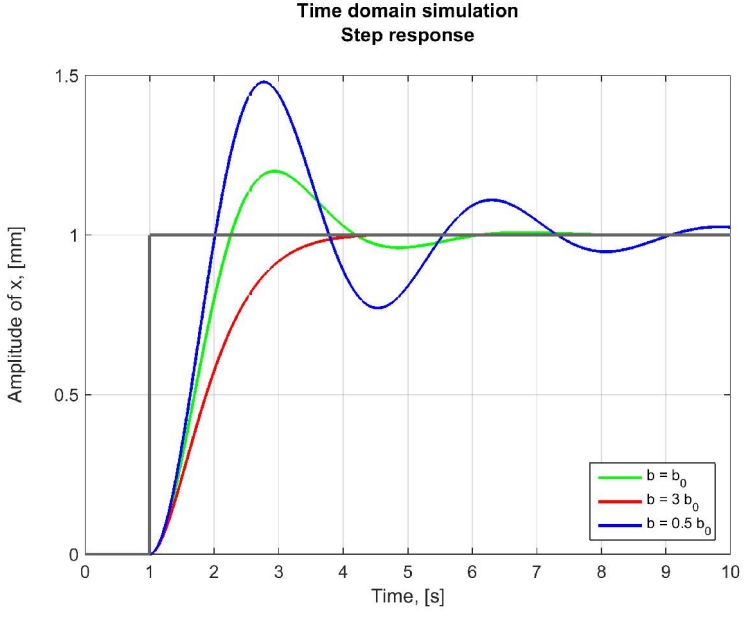
- построение карты нулей и полюсов;
- построение графика корневого годографа;

Переходной процесс

Переходной процесс - это реакция (т.е. изменяется выход $y(t)$) динамической системы на ступенчатое входное воздействие.

По графику переходного процесса определяют:

- устойчивость системы прямым методом;
- основные характеристики качества:
  - величину перерегулирования;
  - время установившегося процесса;
- величину статической ошибки;
- и ряд других второстепенных характеристик.


Time domain simulation
Step response
Amplitude of x, [mm]
Time, [s]
b = b0
b = 3 b0
b = 0.5 b0
0
0.5
1
1.5
0
1
2
3
4
5
6
7
8
9
10

Импульсная переходная характеристика

Импульсная переходная характеристика - это реакция (т.е. изменяется выход $y(t)$) динамической системы на импульсное входное воздействие.

Входное импульсное воздействие для непрерывных систем описывается дельта-функцией Дирака. На практике величина импульса равная $+\infty$ недостижима, поэтому магнитуда импульса обладает конечным значением, как правило, при моделирование магнитуда импульса принимается равной единице.

Частотная характеристика

Частотная характеристика - это в общем случае комплексная функция частоты ω, показывающая как меняется амплитуда и фаза входного сигнала (на частоте ω) при его прохождении через систему.

Как правило, частотная характеристика определяется из описания системы в переменных вход-выход следующим образом:

$$W(j\omega) = W(s)\Big|_{s=j\omega}$$

На практике, речь идет о векторе комплексных значений $W(j\omega)$, вычисленных на заданных частотах в векторе ω.

Передаточная функция:

$$W(s) = \frac{1}{\tau s + 1}$$

Получим частотную характеристику:

$$W(\jmath\omega) = W(s)\Big|_{s=\jmath\omega} = \frac{1}{\tau\jmath\omega + 1} = \frac{(\tau\jmath\omega - 1)}{(\tau\jmath\omega + 1)(\tau\jmath\omega - 1)}$$
$$= \frac{(\tau\jmath\omega - 1)}{(\tau\jmath\omega)^2 - 1^2} = \frac{(\tau\jmath\omega - 1)}{-(\tau\omega)^2 - 1}$$
$$= \frac{1}{(\tau\omega)^2 + 1} - \jmath\frac{\tau\omega}{(\tau\omega)^2 + 1} = U(\omega) + \jmath V(\omega)$$

Подставляя значения ω в рад/сек, получаем вектор значений $W(\jmath\omega)$

Амплитудно-частотная характеристика

Амплитудно-частотная характеристика - это, в общем случае, комплексная функция частоты ω, показывающая как меняется амплитуда входного сигнала (на частоте ω) при его прохождении через систему.

Зная $W(j\omega)$, можно вычислить АЧХ:

$$\text{АЧХ} := \left| W(j\omega) \right| = \sqrt{U(\omega)^2 + V(\omega)^2}$$

Как правило, график АЧХ строится в логарифмическом масштабе, как по частоте, так и по амплитуде. Для этого АЧХ преобразуется следующим образом:

$$\text{АЧХ} := 20 \log_{10} \left(\left| W(j\omega) \right| \right)$$

Фазо-частотная характеристика

Фазо-частотная характеристика - это в общем случае комплексная функция частоты ω, показывающая как меняется фаза входного сигнала (на частоте ω) при его прохождении через систему.

Зная $W(j\omega)$, можно вычислить ФЧХ:

$$\text{ФЧХ} := \arg\left(W(j\omega)\right) = \arctan\left(\frac{U(\omega)}{V(\omega)}\right) + k\pi$$

Как правило, график ФЧХ строится в логарифмическом масштабе по частоте, а фаза отображается в градусах, для этого ФЧХ преобразуется следующим образом:

$$\text{ФЧХ} := \arg\left(W(j\omega)\right) \cdot \frac{180}{\pi}$$

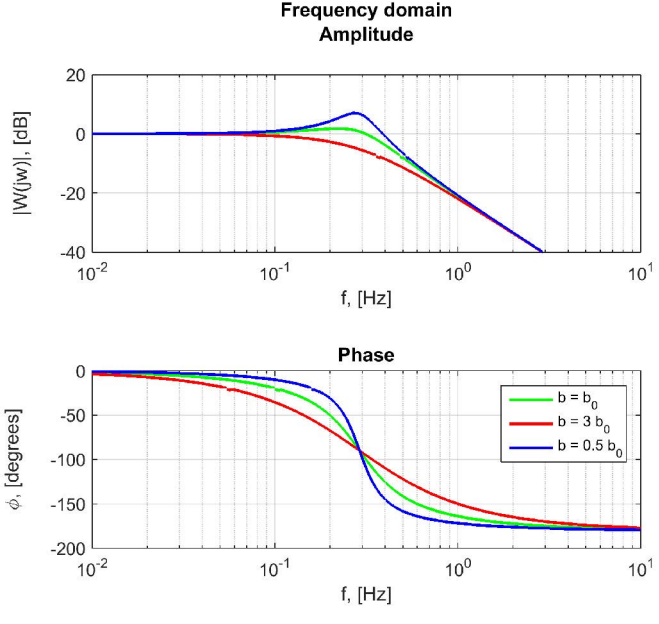
Диаграмма Боде

Диаграмма Боде (логарифмическая амплитудно-фазо-частотная характеристика) - это два графика, один из которых АЧХ, а другой ФЧХ, построенные по описанному ранее алгоритму.

Диаграмма Боде (ЛАФЧХ), по сути, графическое представление частотной характеристики, которое полностью характеризует то, как система изменяет входное воздействие на каждой частоте.

Диаграмма Боде используется для:

- анализа фильтрующих способностей системы;
- определения устойчивости замкнутых систем;
- определение запасов устойчивости по амплитуде и фазе;


Frequency domain
Amplitude
|W([jw)|, [dB]
20
0
-20
-40
10^-2
10^-1
10^0
10^1
f, [Hz]
Phase
ϕ, [degrees]
0
-50
-100
-150
-200
b = b0
b = 3 b0
b = 0.5 b0
f, [Hz]

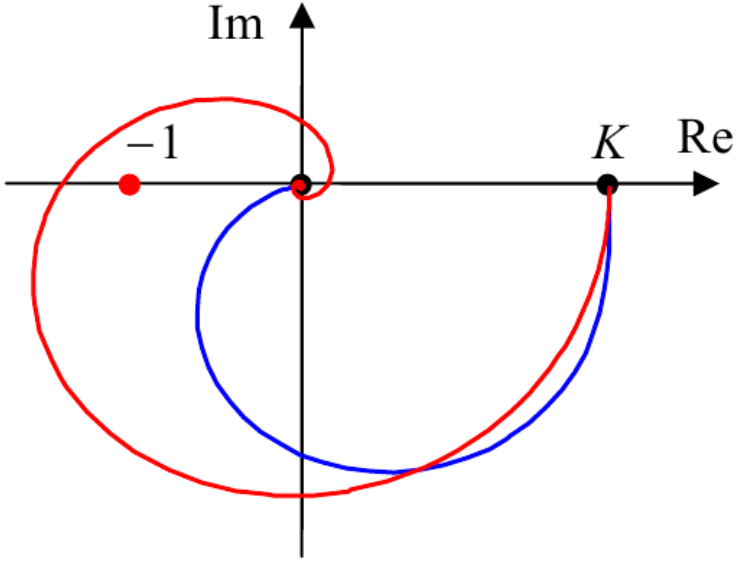
Годограф Найквиста

Годограф Найквиста (амплитудно-фазо-частотная характеристика - АФЧХ) - еще один вариант отображения частотной характеристики, который строится в комплексных осях.

Для каждого значения ω_i получаем одну точку на комплексной плоскости как $[U(\omega_i), V(\omega_i)]$. Соединяя точки от $\omega = 0$ до $\omega = +\infty$, получаем годограф Найквиста.

Годограф Найквиста используется для:

- определения устойчивости замкнутых систем;
- определение запасов устойчивости по амплитуде и фазе;


Im
Re
−1
K

Передаточная функция динамический системы может быть представлена в виде отношения полиномов числителя $N(s)$ и знаменателя $D(s)$ следующим образом:

$$W(s) = \frac{N(s)}{D(s)}$$

Нули системы

Нули системы - это корни характеристического полинома числителя $N(s) = 0$ передаточной функции.

Полюса системы

Полюса системы - это корни характеристического полинома знаменателя $D(s) = 0$ передаточной функции. Именно полюса определяют основную динамику и устойчивость рассматриваемой системы.

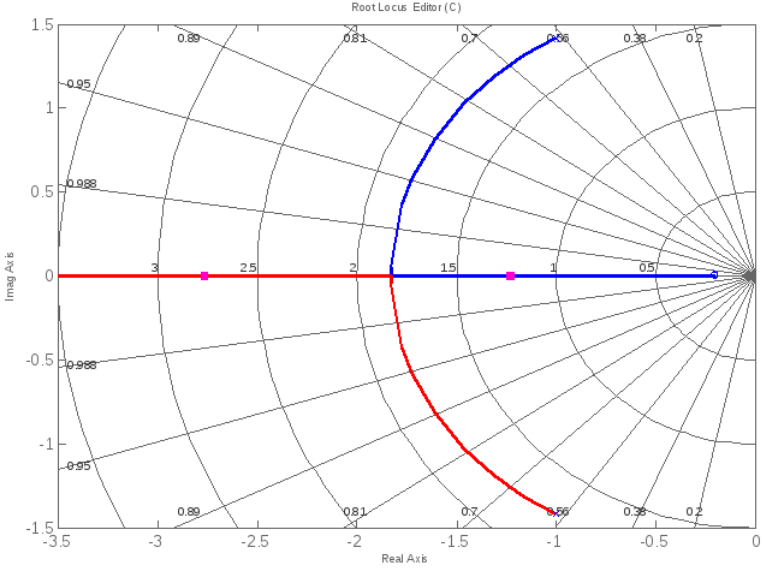
Карта нулей и полюсов системы

Карта нулей и полюсов системы - это графическое отображение значений нулей и полюсов системы на комплексной плоскости.

Корневой годограф

Корневой годограф - это графическое отображение значений нулей и полюсов системы на комплексной плоскости для замкнутой системы, как правило, при изменении коэффициента усиления в цепи обратной связи.

Карта нулей и полюсов - отображает только то, что есть в системе, а корневой годограф показывает как изменяются нули и полюса при изменении, как правило, усиления в цепи обратной связи


Root Locus Editor (C)
Imag Axis
Real Axis
1.5
1
0.5
0
-0.5
-1
-1.5
-3.5
-3
-2.5
-2
-1.5
-1
-0.5
0
0.89
0.81
0.7
0.56
0.38
0.2
0.95
0.988
3
2.5
2
1.5
1
0.5
0

`step`	-	построение переходного процесса;
`impulse`	-	построение импульсной переходной характеристики;
`lsim`	-	моделирование системы;
`freqresp`	-	вычисление частотной характеристики;
`bode`	-	построение диаграммы Боде;
`nyquist`	-	построение годографа Найквиста;
`zero`	-	определение нулей системы;
`pole`	-	определение полюсов системы;
`pzmap`	-	построение карты нулей и полюсов;
`rlocus`	-	построение корневого годографа.

Задача Коши

Заключается в поиске решения дифференциального уравнения, удовлетворяющего заданным начальным условиям.

Для дифференциального уравнения первого порядка:

$$\begin{cases} x^{(1)} = f(x, t) \\ x(t_0) = x_0 \end{cases}$$

Для системы из n дифференциальных уравнений первого порядка:

$$\begin{cases} \mathbf{x}^{(1)} = \mathbf{f}(\mathbf{x}, t) \\ \mathbf{x}(t_0) = \mathbf{x}_0 \end{cases}$$

Дифференциальные уравнения n-ого порядка приводятся к системе n дифференциальных уравнений первого порядка, т.е. к форме Коши

Явные методы:

$$\mathbf{x}_{k+1} = \mathrm{sol}\left(\mathbf{x}_k\right)$$

Неявные методы:

$$\mathbf{x}_{k+1} = \mathrm{sol}\left(\mathbf{x}_{k+1}, \mathbf{x}_k\right)$$

Одношаговые методы:

$$\mathbf{x}_{k+1} = \mathrm{sol}\left(\mathbf{x}_k\right)$$
$$\mathbf{x}_{k+1} = \mathrm{sol}\left(\mathbf{x}_{k+1}, \mathbf{x}_k\right)$$

Многошаговые методы ($p > 0$ шагов):

$$\mathbf{x}_{k+1} = \mathrm{sol}\left(\mathbf{x}_k, \ \ldots \ , \mathbf{x}_{k-p}\right)$$
$$\mathbf{x}_{k+1} = \mathrm{sol}\left(\mathbf{x}_{k+1}, \mathbf{x}_k, \ \ldots \ , \mathbf{x}_{k-p}\right)$$

- одношаговые методы:
  - метод Эйлера (прямой);
  - метод Эйлера-Коши;
  - метод Эйлера-Коши с итерационной обработкой;
  - метод Рунге-Кутты третьего порядка;
  - метод Рунге-Кутты четвертого порядка (классический);
  - метод Рунге-Кутты четвертого порядка (3/8);
- адаптивные методы (одношаговые):
  - метод Рунге-Кутты (4, 5);
  - метод Дормана-Принса;
- многошаговые методы:
  - методы Адамса-Башфорта (явные);
  - методы Адамса-Мультона (неявные);

Система дифференциальных уравнений первого порядка:

$$\begin{cases} \mathbf{x}^{(1)} = \mathbf{f}\left(\mathbf{x}, t\right) \\ \mathbf{x}(t_0) = \mathbf{x}_0 \end{cases}$$

Полагаем, что дискретное время смещается на шаг h:

$$t_{k+1} = t_0 + k\,h, \quad k = 0, 1, \ldots$$

Первая производная аппроксимируется через конечную разность

$$\mathbf{x}^{(1)}(t_k) \approx \frac{\mathbf{x}_{k+1} - \mathbf{x}_k}{h}$$

В итоге исходное уравнение примет вид:

$$\mathbf{x}_{k+1} = \mathbf{x}_k + h\,\mathbf{f}\left(\mathbf{x}_k, t_k\right)$$

Это общий вид численного нахождения решения задачи Коши.
Предыдущий шаг + добавка (результат интегрирования)

Система дифференциальных уравнений первого порядка:

$$\begin{cases} \mathbf{x}^{(1)} = \mathbf{f}\left(\mathbf{x}, t\right) \\ \mathbf{x}(t_0) = \mathbf{x}_0 \end{cases}$$

Явный метод Эйлера-Коши:

$$\widetilde{\mathbf{x}}_{k+1} = \mathbf{x}_k + h\,\mathbf{f}\left(\mathbf{x}_k, t_k\right)$$

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \frac{h}{2}\left[\mathbf{f}\left(\mathbf{x}_k, t_k\right) + \mathbf{f}\left(\widetilde{\mathbf{x}}_{k+1}, t_{k+1}\right)\right]$$

где:

$\widetilde{\mathbf{x}}_{k+1}$ - прогноз (оценка), на которую корректируется $\mathbf{x}_{k+1}$.

Неявный метод Эйлера-Коши:

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \frac{h}{2}\left[\mathbf{f}\left(\mathbf{x}_k, t_k\right) + \mathbf{f}\left(\mathbf{x}_{k+1}, t_{k+1}\right)\right]$$

Метод Эйлера-Коши с итерационной обработкой:

$$\mathbf{x}_{k+1,\,0} = \mathbf{x}_k + h\,\mathbf{f}\left(\mathbf{x}_k\,,\,t_k\right)$$
$$\mathbf{x}_{k+1,\,i} = \mathbf{x}_k + \frac{h}{2}\left[\mathbf{f}\left(\mathbf{x}_k\,,\,t_k\right) + \mathbf{f}\left(\mathbf{x}_{k+1,\,i-1}\,,\,t_{k+1}\right)\right]$$

где:

i - счетчик итераций (нужны единицы).

Улучшенный метод Эйлера:

$$\mathbf{x}_{k+0.5} = \mathbf{x}_k + \frac{h}{2}\,\mathbf{f}\left(\mathbf{x}_k\,,\,t_k\right)$$
$$\mathbf{x}_{k+1} = \mathbf{x}_k + h\,\mathbf{f}\left(\mathbf{x}_{k+0.5}\,,\,t_{k+0.5}\right)$$

Явные методы:

$$\begin{array}{c|cccc} 0 & & & & \\ c_2 & a_{21} & & & \\ \vdots & \vdots & \ddots & & \\ c_s & a_{s1} & \dots & a_{s,s-1} & \\ \hline & b_1 & \dots & b_{s-1} & b_s \end{array}$$

Неявные методы:

$$\begin{array}{c|ccc} c_1 & a_{11} & \dots & a_{1s} \\ c_2 & a_{21} & \dots & a_{2s} \\ \vdots & \vdots & \ddots & \vdots \\ c_s & a_{s1} & \dots & a_{ss} \\ \hline & b_1 & \dots & b_s \end{array}$$

Альтернативная запись:

$$\begin{array}{c|c} \mathbf{c} & \mathbf{A} \\ \hline & \mathbf{b}^\top \end{array}$$

Метод Эйлера (первый порядок):

$$K_1 = \mathbf{f}\left(\mathbf{x}_k\,,\,t_k\right)$$

Решение:

$$\mathbf{x}_{k+1} = \mathbf{x}_k + h\,K_1$$

Таблица Бутчера:

0	0
	1

Метод Ралстона (второй порядок):

$$K_1 = \mathbf{f}\left(\mathbf{x}_k\,,\,t_k\right)$$

$$K_2 = \mathbf{f}\left(\mathbf{x}_k + 2/3\,h\,K_1\,,\,t_k + 2/3\,h\right)$$

Решение:

$$\mathbf{x}_{k+1} = \mathbf{x}_k + h\left(1/4\,K_1 + 3/4\,K_2\right)$$

Таблица Бутчера:

0	0	0
2/3	2/3	0
	1/4	3/4

Явные методы Рунге-Кутты:

$$
\begin{aligned}
K_1 &= \mathbf{f}\left(\mathbf{x}_k\,,\, t_k\right)\,,\\
K_2 &= \mathbf{f}\left(\mathbf{x}_k + a_{21}\,h\,K_1\,,\, t_k + c_2\,h\right)\,,\\
&\vdots\\
K_s &= \mathbf{f}\left(\mathbf{x}_k + a_{s1}\,h\,K_1 + \,\ldots\, + a_{s,s-1}\,h\,K_{s-1}\,,\, t_k + c_s\,h\right)
\end{aligned}
$$

В общем случае вычисление K_i:

$$
K_i = \mathbf{f}\left(\mathbf{x}_k + h\,\sum_{j=1}^{i-1}\,a_{ij}\,K_j\,,\, t_k + c_i\,h\right)
$$

Решение получается в виде:

$$
\mathbf{x}_{k+1} = \mathbf{x}_k + h\,\sum_{i=1}^{s}\,b_i\,K_i
$$

Таблица Бутчера:

0			
1/2	1/2		
1/2	−1	2	
	1/6	2/3	1/6

Начальные условия:

$$\mathbf{x}_0 = \mathbf{x}(t_0)$$

Вычисление коэффициентов:

$$K_1 = \mathbf{f}\left(\mathbf{x}_k\,,\, t_k\right)$$

$$K_2 = \mathbf{f}\left(\mathbf{x}_k + 1/2\,h\,K_1\,,\, t_k + 1/2\,h\right)$$

$$K_3 = \mathbf{f}\left(\mathbf{x}_k - h\,K_1 + 2\,h\,K_2\,,\, t_k + 1/2\,h\right)$$

Решение на каждом шаге:

$$\mathbf{x}_{k+1} = \mathbf{x}_k + h\left(\frac{1}{6}K_1 + \frac{2}{3}K_2 + \frac{1}{6}K_3\right)$$

Таблица Бутчера:

0				
1/2	1/2			
1/2	0	1/2		
1	0	0	1	
	1/6	1/3	1/3	1/6

Начальные условия:

$$\mathbf{x}_0 = \mathbf{x}(t_0)$$

Вычисление коэффициентов:

$$K_1 = \mathbf{f}\left(\mathbf{x}_k\,,\, t_k\right)$$

$$K_2 = \mathbf{f}\left(\mathbf{x}_k + 1/2\,h\,K_1\,,\, t_k + 1/2\,h\right)$$

$$K_3 = \mathbf{f}\left(\mathbf{x}_k + 1/2\,h\,K_2\,,\, t_k + 1/2\,h\right)$$

$$K_4 = \mathbf{f}\left(\mathbf{x}_k + h\,K_3\,,\, t_k + h\right)$$

Решение на каждом шаге:

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \frac{h}{6}\left(K_1 + 2\,K_2 + 2\,K_3 + K_4\right)$$

Таблица Бутчера:

0				
1/3	1/3			
2/3	−1/3	1		
1	1	−1	1	
	1/8	3/8	3/8	1/8

Начальные условия:

$$\mathbf{x}_0 = \mathbf{x}(t_0)$$

Вычисление коэффициентов:

$$K_1 = \mathbf{f}\left(\mathbf{x}_k\,,\, t_k\right)$$

$$K_2 = \mathbf{f}\left(\mathbf{x}_k + 1/3\,h\,K_1\,,\, t_k + 1/3\,h\right)$$

$$K_3 = \mathbf{f}\left(\mathbf{x}_k + \left(K_2 - 1/3\,K_1\right)h\,,\, t_k + 2/3\,h\right)$$

$$K_4 = \mathbf{f}\left(\mathbf{x}_k + (K_1 - K_2 + K_3)\,h\,,\, t_k + h\right)$$

Решение на каждом шаге:

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \frac{h}{8}\left(K_1 + 3\,K_2 + 3\,K_3 + K_4\right)$$

Неявные методы Рунге-Кутты:

$$K_i = \mathbf{f}\left(\mathbf{x}_k + \sum_{j=1}^{s} a_{ij}\, h\, K_j\,,\; t_k + c_i\, h\right)$$

$$\mathbf{x}_{k+1} = \mathbf{x}_k + h \sum_{i=1}^{s} b_i\, K_i$$

Отличия от явных методов:

- матрица коэффициентов полная, а не нижняя треугольная;
- на каждом шаге решаем систему уравнений.

Условие применимости:

$$\sum_{j=1}^{i-1} a_{ij} = c_i \quad \text{для} \quad i = 2, \ldots, s$$

Метод Radau IA 3-ого порядка:

$$\begin{array}{c|cc} 0 & \frac{1}{4} & -\frac{1}{4} \\ \frac{2}{3} & \frac{1}{4} & \frac{5}{12} \\ \hline & \frac{1}{4} & \frac{3}{4} \end{array}$$

Метод Radau IIA 3-ого порядка:

$$\begin{array}{c|cc} \frac{1}{3} & \frac{5}{12} & -\frac{1}{12} \\ 1 & \frac{3}{4} & \frac{1}{4} \\ \hline & \frac{3}{4} & \frac{1}{4} \end{array}$$

Метод Radau IA 5-ого порядка:

$$
\begin{array}{c|ccc}
0 & \frac{1}{9} & \frac{-1-\sqrt{6}}{18} & \frac{-1+\sqrt{6}}{18} \\
\frac{3}{5}-\frac{\sqrt{6}}{10} & \frac{1}{9} & \frac{11}{45}+\frac{7\sqrt{6}}{360} & \frac{11}{45}-\frac{43\sqrt{6}}{360} \\
\frac{3}{5}+\frac{\sqrt{6}}{10} & \frac{1}{9} & \frac{11}{45}+\frac{43\sqrt{6}}{360} & \frac{11}{45}-\frac{7\sqrt{6}}{360} \\
\hline
 & \frac{1}{9} & \frac{4}{9}+\frac{\sqrt{6}}{36} & \frac{4}{9}-\frac{\sqrt{6}}{36}
\end{array}
$$

Метод Radau IIA 5-ого порядка:

$\frac{2}{5} - \frac{\sqrt{6}}{10}$	$\frac{11}{45} - \frac{7\sqrt{6}}{360}$	$\frac{37}{225} - \frac{169\sqrt{6}}{1800}$	$-\frac{2}{225} + \frac{\sqrt{6}}{75}$
$\frac{2}{5} + \frac{\sqrt{6}}{10}$	$\frac{37}{225} + \frac{169\sqrt{6}}{1800}$	$\frac{11}{45} + \frac{7\sqrt{6}}{360}$	$-\frac{2}{225} - \frac{\sqrt{6}}{75}$
1	$\frac{4}{9} - \frac{\sqrt{6}}{36}$	$\frac{4}{9} + \frac{\sqrt{6}}{36}$	$\frac{1}{9}$
	$\frac{4}{9} - \frac{\sqrt{6}}{36}$	$\frac{4}{9} + \frac{\sqrt{6}}{36}$	$\frac{1}{9}$

Метод простой итерации

Применяется для поиска решения в неявных методах Рунге-Кутты:

$$\mathbf{K} = \begin{bmatrix} K_1 & K_2 & \cdots & K_s \end{bmatrix}$$

На каждой k-ой итерации включается вложенный итерационный процесс вычисления $\mathbf{K}$, т.е.:

$$\mathbf{K}_{p+1} = \mathbf{f}\left(\mathbf{K}_p\right)$$

который продолжается пока:

$$\left\|\mathbf{K}_{p+1} - \mathbf{K}_p\right\| > \varepsilon \qquad \text{и} \qquad p < p_{\max}$$

Начальное приближение $\mathbf{K}_0$ на каждом k-ом шаге берется как решение $\mathbf{K}$, полученное на предыдущем $k-1$ шаге.

Адаптивные численные методы решения задачи Коши

Адаптивным численным методом решения задачи коши называется любой метод в котором значение шага h изменяется внутри самого метода для повышения точности вычислений.

Существуют два принципиальных подхода реализации адаптианых методов:

- При вычислении $\mathbf{x}_{k+1}$ шаг h постоянно уменьшается до h^*, пока не будет достигнута требуемая точность. Следующий шаг начинается с момента времени $t_k + h^*$.
  (перерасчет решения на каждом шаге)
- При вычислении $\mathbf{x}_{k+1}$ определяется удовлетворяет ли решение заданной точности и если нет, то шаг h для следуюшей итерации, $\mathbf{x}_{k+2}$ уменьшается, а значение $\mathbf{x}_{k+1}$ принимается таким, каким есть.
  (без перерасчета решения на каждом шаге)

Как правило, если точность удовлетворительная, то в методах предусматривается увеличение шага в целях повышения быстродействия.

Контроль точности на каждом шаге:

- вычисляем $\mathbf{x}_{k+1}$ с шагом h и с шагом $2h$;
- оцениваем погрешность решения как:

$$E = \frac{\left\| \mathbf{x}_{k+1}^{h} - \mathbf{x}_{k+1}^{2h} \right\|}{2^s - 1}$$

- при $E < \varepsilon$ продолжаем вычисления с тем же шагом;
- при $E \geqslant \varepsilon$ уменьшаем шаг h и повторяем вычисления с t_k.

Грубый контроль точности (для метода 4-го порядка) на k-ом шаге:

$$\theta^k = \left\| \frac{K_2 - K_3}{K_1 - K_2} \right\|$$

$\theta^k > 0.01$ - уменьшаем h,
$\theta^k < 0.001$ - увеличиваем h.

Таблица Бутчера:

0						
1/4	1/4					
3/8	3/32	9/32				
12/13	1932/2197	−7200/2197	7296/2197			
1	439/216	−8	3680/513	−845/4104		
1/2	−8/27	2	−3544/2565	1859/4104	−11/40	
	16/135	0	6656/12825	28561/56430	−9/50	2/55
	25/216	0	1408/2565	2197/4104	−1/5	0

На одной итерации считаются два решения, их разница сравнивается с ε, затем принимается решение об изменении шага h, до те пор, пока разница решений не удовлетворит точности

Таблица Бутчера:

0							
$1/5$	$1/5$						
$3/10$	$3/40$	$9/40$					
$4/5$	$44/45$	$-56/15$	$32/9$				
$8/9$	$19372/6561$	$-25360/2187$	$64448/6561$	$-212/729$			
1	$9017/3168$	$-355/33$	$46732/5247$	$49/176$	$-5103/18656$		
1	$35/384$	0	$500/1113$	$125/192$	$-2187/6784$	$11/84$	
	$35/384$	0	$500/1113$	$125/192$	$-2187/6784$	$11/84$	0
	$5179/57600$	0	$571/16695$	$393/640$	$-92097/339200$	$187/2100$	$1/40$

```
function [X, T] = ode(f, t_begin, t_end, h0, eps1, eps2)
    ... инициализация X, T, k = 1, h = h0 ...
    t_k = t_begin;
    while t_k < t_end
        ... вычисление K_i, x_1, x_2 ...
        t_k = t_k + h;
        X(:, k+1) = x_2;
        T(k+1) = t_k;
        k = k + 1;
        if norm(x_1 - x_2) > eps1
            h = h / 2;
        elseif norm(x_1 - x_2) < eps2
            h = h * 2;
        end
    end
end
```

```
function [X, T] = ode(f, t_begin, t_end, h0, eps1, eps2)
    ... инициализация X, T, k = 1, h = h0 ...
    t_k = t_begin;
    while t_k < t_end
        ... вычисление K_i, x_1, x_2 ...
        if norm(x_1 - x_2) > eps1
            h = h / 2;
        else
            t_k = t_k + h;
            X(:, k+1) = x_2;
            T(k+1) = t_k;
            k = k + 1;
            if norm(x_1 - x_2) < eps2
                h = h * 2;
            end
        end
    end
end
```

```
function [X, T] = ode(f, t_begin, t_end, h0, eps1, eps2)
    ...
    while t_k < t_end
        if t_k + h > t_end
            h = t_end - t_k;
        end
        ...
    end
end
```

Экстраполяционные методы (Адамса-Башфорта, явные):

$$\mathbf{x}_{k+1} = \mathbf{x}_k + h \sum_{\lambda=0}^{k} u_{-\lambda}\, \mathbf{f}\left(\mathbf{x}_{k-\lambda}\,,\, t_{k-\lambda}\right)$$

В случае методов Адамса-Башфорта количество переменных K_i определяется порядком метода, равным количеству точек решения, которые необходимо предварительно расчитать (включая начальные условия).

Интерполяционные методы (Адамса-Мультона, неявные):

$$\mathbf{x}_{k+1} = \mathbf{x}_k + h \sum_{\lambda=-1}^{k-1} v_{-\lambda}\, \mathbf{f}\left(\mathbf{x}_{k-\lambda}\,,\, t_{k-\lambda}\right)$$

Методы Адамса-Башфорта:

$$\mathbf{x}_{k+1} = \mathbf{x}_k + h\,\mathbf{f}\left(\mathbf{x}_k\,,\,t_k\right)$$

$$\mathbf{x}_{k+2} = \mathbf{x}_{k+1} + h\left(\frac{3}{2}\mathbf{f}\left(\mathbf{x}_{k+1}\,,\,t_{k+1}\right) - \frac{1}{2}\mathbf{f}\left(\mathbf{x}_k\,,\,t_k\right)\right)$$

$$\mathbf{x}_{k+3} = \mathbf{x}_{k+2} + h\left(\frac{23}{12}\mathbf{f}\left(\mathbf{x}_{k+2}\,,\,t_{k+2}\right) - \frac{16}{12}\mathbf{f}\left(\mathbf{x}_{k+1}\,,\,t_{k+1}\right) + \frac{5}{12}\mathbf{f}\left(\mathbf{x}_k\,,\,t_k\right)\right)$$

$$\mathbf{x}_{k+4} = \mathbf{x}_{k+3} + h\left(\frac{55}{24}\mathbf{f}\left(\mathbf{x}_{k+3}\,,\,t_{k+3}\right) - \frac{59}{24}\mathbf{f}\left(\mathbf{x}_{k+2}\,,\,t_{k+2}\right) + \frac{37}{24}\mathbf{f}\left(\mathbf{x}_{k+1}\,,\,t_{k+1}\right) - \frac{9}{24}\mathbf{f}\left(\mathbf{x}_k\,,\,t_k\right)\right)$$

Начальные условия:

$$\mathbf{x}_0 = \mathbf{x}(t_0)$$

Вычисление коэффициентов:

$$K_1 = 1901/720 \cdot \mathbf{f}\left(\mathbf{x}_{k+4}, t_{k+4}\right)$$
$$K_2 = -1387/360 \cdot \mathbf{f}\left(\mathbf{x}_{k+3}, t_{k+3}\right)$$
$$K_3 = 109/30 \cdot \mathbf{f}\left(\mathbf{x}_{k+2}, t_{k+2}\right)$$
$$K_4 = -637/360 \cdot \mathbf{f}\left(\mathbf{x}_{k+1}, t_{k+1}\right)$$
$$K_5 = 251/720 \cdot \mathbf{f}\left(\mathbf{x}_{k+0}, t_{k+0}\right)$$

Решение на каждом шаге:

$$\mathbf{x}_{k+5} = \mathbf{x}_{k+4} + h\left(K_1 + K_2 + K_3 + K_4 + K_5\right)$$

Первые $n - 1$ значений $\mathbf{f}$ можно вычислить любым явным методом

Примеры методов Адамса-Мультона:

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \frac{1}{2} h \left(\mathbf{f}\left(\mathbf{x}_{k+1}, t_{k+1} \right) + \mathbf{f}\left(\mathbf{x}_k, t_k \right) \right)$$

$$\mathbf{x}_{k+2} = \mathbf{x}_{k+1} + h \left(\frac{5}{12} \mathbf{f}\left(\mathbf{x}_{k+2}, t_{k+2} \right) + \frac{2}{3} \mathbf{f}\left(\mathbf{x}_{k+1}, t_{k+1} \right) - \frac{1}{12} \mathbf{f}\left(\mathbf{x}_k, t_k \right) \right)$$

$$\mathbf{x}_{k+3} = \mathbf{x}_{k+2} + h \left(\frac{3}{8} \mathbf{f}\left(\mathbf{x}_{k+3}, t_{k+3} \right) + \frac{19}{24} \mathbf{f}\left(\mathbf{x}_{k+2}, t_{k+2} \right) - \frac{5}{24} \mathbf{f}\left(\mathbf{x}_{k+1}, t_{k+1} \right) + \frac{1}{24} \mathbf{f}\left(\mathbf{x}_k, t_k \right) \right)$$

$$\mathbf{x}_{k+4} = \mathbf{x}_{k+3} + h \left(\frac{251}{720} \mathbf{f}\left(\mathbf{x}_{k+4}, t_{k+4} \right) \frac{646}{720} \mathbf{f}\left(\mathbf{x}_{k+3}, t_{k+3} \right) - \frac{264}{720} \mathbf{f}\left(\mathbf{x}_{k+2}, t_{k+2} \right) + \frac{106}{720} \mathbf{f}\left(\mathbf{x}_{k+1}, t_{k+1} \right) - \frac{19}{720} \mathbf{f}\left(\mathbf{x}_k, t_k \right) \right)$$

Метод прогноза-коррекции

Применяется для поиска решения в методах Адамса-Мультона. Суть метода в формировании прогноза неизвестного значения $\mathbf{x}_{k+1}$ и последующей коррекции уже при непосредственном вычислении $\mathbf{x}_{k+1}$, т.е., например для метода Адамса-Мультона 2 порядка:

Прогноз:

$$\hat{\mathbf{x}}_{k+2} = \mathbf{x}_{k+1} + h \left(\frac{3}{2} \mathbf{f}\left(\mathbf{x}_{k+1}, t_{k+1} \right) - \frac{1}{2} \mathbf{f}\left(\mathbf{x}_{k}, t_{k} \right) \right)$$

Коррекция:

$$\mathbf{x}_{k+2} = \mathbf{x}_{k+1} + h \left(\frac{5}{12} \mathbf{f}\left(\hat{\mathbf{x}}_{k+2}, t_{k+2} \right) + \frac{2}{3} \mathbf{f}\left(\mathbf{x}_{k+1}, t_{k+1} \right) - \frac{1}{12} \mathbf{f}\left(\mathbf{x}_{k}, t_{k} \right) \right)$$

Метод прогноза-коррекции

Существует несколько схем прогноза-коррекции, в том числе и с использованием итерационного процесса.

Коррекция по правилу трапеций (требуется вывод формул корректора для более оптимального количества вычислений):

$$\mathbf{x}_{k+2} = \mathbf{x}_{k+1} + h\,\frac{\mathbf{f}\left(\mathbf{x}_k\,,\,t_k\right) + \mathbf{f}\left(\hat{\mathbf{x}}_{k+2}\,,\,t_{k+2}\right)}{2}$$

A-устойчивость

Численный метод решения задачи Коши A-устойчив, если при фиксированном шаге h для дифференциального уравнения

$$\dot{x} = k\,x\,, \qquad \text{при} \qquad x(0) = 1 \qquad \text{и} \qquad \Re(k) < 0$$

получаемое решение стремится к нулю при $t \to \infty$. Другими словами A-устойчивость - абсолютная устойчивость.

Функция устойчивости

Для методов Рунге-Кутты функция устойчивости имеет вид:

$$\varphi(z) = \frac{\det \mathbf{I} - z\,\mathbf{A} + z\,\mathbf{e}\,\mathbf{b}^{\top}}{\det \mathbf{I} - z\,\mathbf{A}}$$

где $\mathbf{e}$ - единичный вектор.

L-устойчивость

Численный метод решения задачи Коши L-устойчив, если он A-устойчив (кроме метода трапеций) и:

$$|\varphi(z)| \to 0 \qquad \text{при} \qquad z \to \infty$$

`ode45` - использует алгоритм Dormand-Prince;

`ode23` - использует алгоритм Bogacki–Shampine;

`ode113` - многошаговые методы Адамса (явные и неявные).

Численное дифференцирование

Метод определения значений производной n-ого порядка заданной функции, в основе которого лежит аппроксимация производной на небольшом участке и который на практике используется с целью, исключения сложных аналитических вычислений и уменьшения объема вычислений заданной аналитически функции.

Как правило, это требуется при решении задач оптимизации, где используются специальные оптимизированные алгоритмы, которые, по факту, базируются на методе конечных разностей.

Самый простой вариант аппроксимации производной - по двум точкам, который может быть решен через прямые, обратные (в обоих случаях - первый порядок точности) и центральные (второй порядок точности) конечные разности.

Аппроксимация производной прямой конечной разностью:

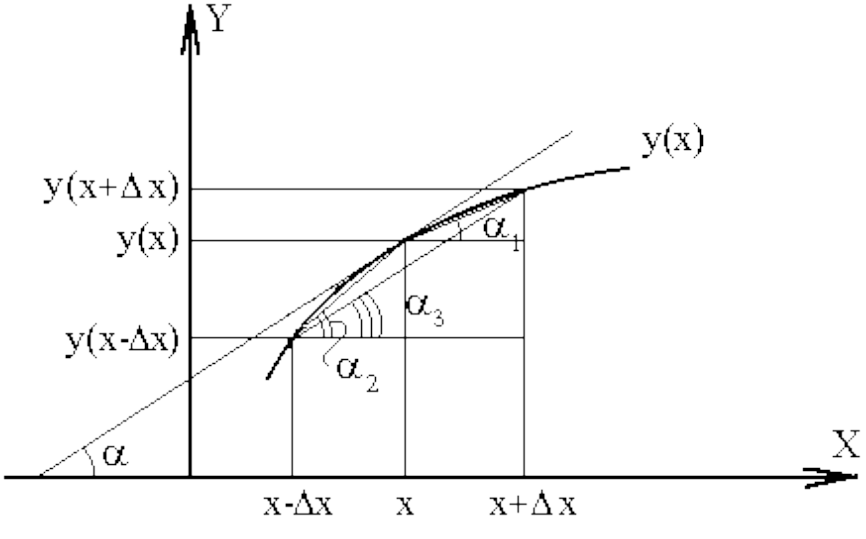
$$f^{(1)}(x) \approx \frac{f(x+h) - f(x)}{h}$$

Аппроксимация производной обратной конечной разностью:

$$f^{(1)}(x) \approx \frac{f(x) - f(x-h)}{h}$$

Аппроксимация производной центральной конечной разностью:

$$f^{(1)}(x) \approx \frac{f(x+h) - f(x-h)}{2h}$$


Y
y(x)
y(x+Δx)
y(x)
y(x-Δx)
α1
α3
α2
α
X
x-Δx
x
x+Δx

Повышение точности возможно с увеличением количества точек. При этом больше точек - больше вычислений. Реализуется также через прямые, обратные или центральные разности.

Аппроксимация с 4-ым порядком точности:

$$f^{(1)}(x) \approx \frac{-f(x+2h) + 8f(x+h) - 8f(x-h) + f(x-2h)}{12h}$$

Аппроксимация с 6-ым порядком точности:

$$f^{(1)}(x) \approx \frac{1}{h}\left[\frac{1}{60}\left[-f(x-3h) + f(x+3h)\right] + \frac{3}{20}\left[f(x-2h) - f(x+2h)\right] + \frac{3}{4}\left[-f(x-h) + f(x+h)\right]\right]$$

Аппроксимация со 2-ым порядком точности:

$$f^{(2)}(x) \approx \frac{f(x-h) - 2\,f(x) + f(x+h)}{h^2}$$

Аппроксимация с 4-ым порядком точности:

$$f^{(2)}(x) \approx \frac{-f(x-2\,h) + 16\,f(x-h) - 30\,f(x) + 16\,f(x+h) - f(x+2\,h)}{12\,h^2}$$

Вывод формул описан в статье:

Generation of Finite Difference Formulas on Arbitrarily Spaced Grids.
Bengt Fornberg
Mathematics of Computation. Vol. 51, #184, pp. 699-706

Аппроксимация со 2-ым порядком точности:

$$f^{(1)}(x) \approx \frac{f(x+h) - f(x-h)}{2h}$$

Аппроксимация с 4-ым порядком точности:

$$f^{(1)}(x) \approx \frac{-f(x+2h) + 8f(x+h) - 8f(x-h) + f(x-2h)}{12h}$$

Возьмем в качестве $f(x) = \sin(x)$ и найдем производную двумя способами:

$f(x)$	0.00	0.58	0.95	0.95	0.58	0.00	-0.58	-0.95	-0.95	-0.58	0.00
$f^{(1)}(x)$		-0.35	-0.57	-0.57	-0.35	0.00	0.35	0.57	0.57	0.35	
$f^{(1)}(x)$			-0.59	-0.59	-0.36	0.00	0.36	0.59	0.59		

Это наглядный пример того, что при использовании разностных схем вычислений, объем получаемых данных меньше, чем объем исходных данных

Матрица Якоби:

Матрица Якоби **J** - матрица первых частных производных многомерной векторной функции:

$$\mathbf{J} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \cdots & \frac{\partial f_2}{\partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f_m}{\partial x_1} & \frac{\partial f_m}{\partial x_2} & \cdots & \frac{\partial f_m}{\partial x_n} \end{bmatrix}$$

Вычисление матрицы Якоби требуется при решении практически всех задач оптимизации, как правило, на каждой итерации

В целом подход вычисления матрицы Якоби аналогичен вычислению производной скалярной функции, т.е. через аппроксимацию производной (в данном случае по каждому аргументу многомерной функции) с использованием конечных разностей.

С использованием прямых конечных разностей:

$$J_{ij} = \frac{f_i\left(x_1, x_2, \ldots, x_j + h, \ldots, x_n\right) - f_i\left(\mathbf{x}\right)}{h}$$

С использованием центральных конечных разностей:

$$J_{ij} = \frac{f_i\left(x_1, \ldots, x_j + h, \ldots, x_n\right) - f_i\left(x_1, \ldots, x_j - h, \ldots, x_n\right)}{2h}$$

Для прямых конечных разностей $\mathbf{f}(\cdot)$ вычисляется в $m(n+1)$ точках. Для центральных конечных разностей $\mathbf{f}(\cdot)$ вычисляется в $2mn$ точках.

Метод Бройдена

Метод численного определения матрицы Якоби в ходе итерационного процесса решения оптимизационной задачи. В этом методе на первой итерации $\mathbf{J}$ вычисляется либо по ранее полученной аналитической форме, либо по методу конечных разностей. На последующих шагах итерационного процесса $\mathbf{J}$ не вычисляется непосредственно, а корректируется согласно следующей формуле:

$$\mathbf{J}_k = \mathbf{J}_{k-1} + \frac{\mathbf{f}\left(\mathbf{x}_k\right) - \mathbf{f}\left(\mathbf{x}_{k-1}\right) - \mathbf{J}_{k-1}\left(\mathbf{x}_k - \mathbf{x}_{k-1}\right)}{\left\| \left(\mathbf{x}_k - \mathbf{x}_{k-1}\right) \right\|^2} \left(\mathbf{x}_k - \mathbf{x}_{k-1}\right)^\top$$

Данный метод позволяет избежать многократных вычислений $\mathbf{f}(\cdot)$

Матрица Гессе:

Матрица Гессе **H** - матрица вторых частных производных одномерной функции $f(\mathbf{x})$:

$$\mathbf{H} = \begin{bmatrix} \dfrac{\partial^2 f}{\partial x_1^2} & \dfrac{\partial^2 f}{\partial x_1 \partial x_2} & \cdots & \dfrac{\partial^2 f}{\partial x_1 \partial x_n} \\ \dfrac{\partial^2 f}{\partial x_2 \partial x_1} & \dfrac{\partial^2 f}{\partial x_2^2} & \cdots & \dfrac{\partial^2 f}{\partial x_2 \partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \dfrac{\partial^2 f}{\partial x_n \partial x_1} & \dfrac{\partial^2 f}{\partial x_n \partial x_2} & \cdots & \dfrac{\partial^2 f}{\partial x_n^2} \end{bmatrix}$$

Вычисление матрицы Гессе требуется при решении многих задач оптимизации, как правило, в ходе итерационного процесса.

Метод Бройдена-Флетчера-Гольдфарба-Шанно

Метод численного определения матрицы Гессе в ходе итерационного процесса решения оптимизационной задачи. В этом методе начальное значение $\mathbf{H}$ принимается любой хорошо обусловленной матрицей (например, единичной), после чего на каждой итерации значение матрицы Гессе корректируется следующим соотношением (на примере прямых конечных разностей):

$$\mathbf{H}_{k+1} = \mathbf{H}_k + \frac{\mathbf{H}_k \, \mathbf{s}_k \, \mathbf{s}_k^\top \, \mathbf{H}_k}{\mathbf{s}_k^\top \, \mathbf{H}_k \, \mathbf{s}_k} + \frac{\mathbf{y}_k \, \mathbf{y}_k^\top}{\mathbf{y}_k^\top \, \mathbf{s}_k}$$

где

$$\mathbf{s}_k = \mathbf{x}_{k+1} - \mathbf{x}_k$$
$$\mathbf{y}_k = \nabla f\left(\mathbf{x}_{k+1}\right) - \nabla f\left(\mathbf{x}_k\right)$$

для численных вычислений:

`diff` - вычисление правых конечных разностей;
`gradient` - метод центральных конечных разностей.

для аналитических вычислений:

`jacobian` - вычисление матрицы Якоби;
`hessian` - вычисление матрицы Гессе.

Задача оптимизации

Решение некоторой технической задачи, которая требует нахождения минимума (реже максимума) целевой функции J при заданных ограничениях $g(x, u)$ на фазовые переменные x и управление u.

Некоторые виды технических задач с оптимизацией:

- формирование оптимального закона управления;
- аппроксимация экспериментальных данных;
- классификация и кластеризация данных;
- идентификация моделей систем управления.

Целевая функция J формируется определенным образом для применения определенного алгоритма поиска минимума/максимума

Поиск экстремума функции

Задача поиска экстремума (минимум или максимум) функции $f(x)$, определенной (рассматриваемой) на интервале $[a, b]$, заключается в применении определенного алгоритма, результатом которого является минимальное значение функции $f_{\min}(x)$ и соответствующее значение аргумента $x_{\min}$. Данная задача формулируется математически следующим образом:

$$x = \underset{x \in \mathcal{X}}{\operatorname{argmin}} f(x) \qquad \text{или} \qquad x = \underset{x \in \mathcal{X}}{\operatorname{argmax}} f(x)$$

где:

$\mathcal{X}$ - множество возможных значений аргумента x.

по типу поиска x:

- локальные;
- глобальные.

по характеру задачи оптимизации:

- детерминированные;
- стохастические;
- смешанные.

по требованию гладкости $f(x)$ и существованию производных:

- прямые;
- первого порядка - требуется знание $\nabla f(x)$;
- второго порядка - требуется знание $\nabla^2 f(x)$.

- одномерные:
  - метод полного перебора;
  - метод дихотомии;
  - метод золотого сечения.
- прямые:
  - метод Гаусса;
  - метод Хука-Дживса.
- первого порядка - требуется знание $\nabla f(x)$:
  - метод покоординатного спуска;
  - метод наискорейшего спуска (градиентный).
- второго порядка - требуется знание $\nabla^2 f(x)$:
  - метод Ньютона;
  - метод Ньютона-Рафсона.

Метод полного перебора

Заключается в последовательном вычислении полного набора значений функции $f(x)$ на задаваемой сетке из n разбиений аргумента x в интервале $[a, b]$, после чего из полученного набора выбирается минимальное значение $f(x)$ и соответствующее значение аргумента $x_{\min}$. На практике сохранение минимального значения функции и соответствующего аргумента происходит в процессе прохода по сетке.

$$x_k = a + k\,\frac{b-a}{n+1}$$

$$f_{\min}(x) = \Big(f(x_k) \;<\; f_{\min}(x)\Big) \;?\; f(x_k) \;:\; f_{\min}(x)$$

Недостаток метода - большое количество бесполезных вычислений
Точность метода определяется числом разбиений интервала $[a, b]$

Метод дихотомии

Заключается в последовательном приближении к минимуму, где на каждой итерации интервал поиска уменьшается в два раза.

На каждой итерации вычисляем:

$$x_{\min} = \frac{a+b}{2}$$

$$\delta \in \left(0, \frac{b-a}{2}\right)$$

$$\text{если} \quad f(x_{\min} - \delta) \geqslant f(x_{\min} + \delta) \implies a = x_{\min} \implies [x, b]$$

$$\text{иначе} \implies b = x_{\min} \implies [a, x]$$

Критерий остановки алгоритма:

$$|b - a| < \varepsilon$$

Начальные условия:

$$f(x) = (x-1)^2\,, \quad [a,b] = [0.0, 1.5]\,, \quad \varepsilon = 0.1\,, \quad \delta = (b-a)/4$$

Таблица решения:

a	b	δ	x	$f(x-\delta)$		$f(x+\delta)$	$\lvert b-a\rvert$
0	1.5000	0.3750	0.7500	0.3906	$>$	0.0156	0.7500
0.7500	1.5000	0.1875	1.1250	0.0039	$<$	0.0977	0.3750
0.7500	1.1250	0.0938	0.9375	0.0244	$>$	0.0010	0.1875
0.9375	1.1250	0.0469	1.0313	0.0002	$<$	0.0061	0.0938

Полученное решение:

$$x_{\min} = (0.9375 + 1.0313)/2 = 0.9844\,, \qquad f(x_{\min}) = 0.0002$$

Метод золотого сечения

Заключается в последовательном приближении к минимуму, где на каждой итерации интервал поиска делится пропорционально золотому сечению φ:

$$\varphi = \frac{1+\sqrt{5}}{2} \approx 1.62$$

Критерий остановки алгоритма:

$$|b-a| < \varepsilon$$

Итоговое значение аргумента в точке экстремума:

$$x_{\min} = \frac{a+b}{2}$$

Меньше вычислений по сравнению с методом дихотомии

Предварительные расчеты:

$$x_1 = b - \frac{b-a}{\varphi}, \quad x_2 = a + \frac{b-a}{\varphi}$$

На каждой итерации считаем:

$$y_1 = f(x_1), \quad y_2 = f(x_2)$$

После чего проверяем условие, если $y_1 \geqslant y_2$:

$$a = x_1, \quad x_1 = x_2, \quad x_2 = a + \frac{b-a}{\varphi}$$

иначе:

$$b = x_2, \quad x_2 = x_1, \quad x_1 = b - \frac{b-a}{\varphi}$$

Наиболее оптимальный вариант включает перерасчет только одного y

Начальные условия:

$$f(x) = (x-1)^2\,, \quad [a,b] = [0.4, 1.5]\,, \quad \varepsilon = 0.1$$

Таблица решения:

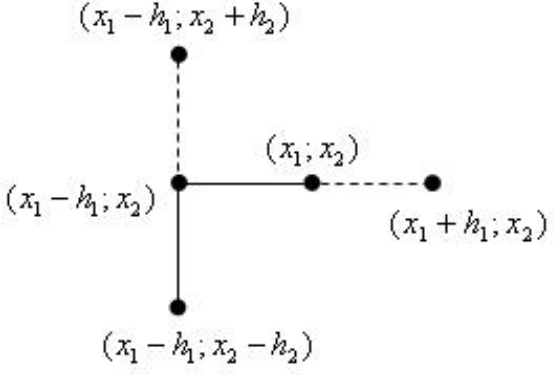
a	b	x_1	x_2	$f(x_1)$		$f(x_2)$	$\lvert b-a\rvert$
0.4000	1.5000	0.8202	1.0798	0.0323	$>$	0.0064	0.6798
0.8202	1.5000	1.0798	1.2403	0.0064	$<$	0.0578	0.4202
0.8202	1.2403	0.9807	1.0798	0.0004	$<$	0.0064	0.2597
0.8202	1.0798	0.9193	0.9807	0.0065	$>$	0.0004	0.1605
0.9193	1.0798	0.9807	1.0185	0.0004	$>$	0.0003	0.0992

Полученное решение:

$$x_{\min} = (0.9807 + 1.0798)/2 = 1.0302\,, \qquad f(x_{\min}) = 9.1469 \times 10^{-4}$$

Метод Хука-Дживса

На n-мерной сетке вычисляются значения функции в точках, отстающих от заданной базовой точки на величину λ, последовательно по каждой координате с перемещением в точку с минимальным значением функции. λ для кажого x_i постепенно уменьшается.



Метод Гаусса

В общем случае сформулированы для **x**, являющегося вектором. Поиск минимума (n-мерной функции) осуществляется итерационным подходом последовательно по каждой компоненте:

$$x_{i,\,k+1} = x_{i,\,k} + \lambda_{i,\,k}\,\mathrm{e}_i$$

где e_i - формируют ортонормированный базис, а $\lambda_{i,\,k}$ находится решением своей оптимизационной задачи:

$$\lambda_{i,\,k} = \underset{\lambda_{i,\,k}}{\operatorname{argmin}}\, f\left(x_{i,\,k} + \lambda_{i,\,k}\,\mathrm{e}_i\right)$$

Методы многомерной оптимизации решаются итерационным подходом

Методы первого порядка

Методы n-мерной оптимизации, где направление поиска экстремума определяется величиной градиента функции:

для метода покоординатного спуска:

$$x_{i,\,k+1} = x_{i,\,k} + \lambda_{i,\,k} \nabla f_i \left(\mathbf{x}_k \right)$$

для градиентного метода:

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \boldsymbol{\lambda}_k \nabla f \left(\mathbf{x}_k \right)$$

где:

$\boldsymbol{\lambda}_k$ - шаг перехода в сторону минимума;
$\nabla f \left(\mathbf{x}_k \right)$ - вектор градиентов минимизируемой функции.

Вектор градиентов определяется как:

$$\nabla f\left(\mathbf{x}_k\right) = -\frac{\partial f(\mathbf{x}_k)}{\partial x_i}$$

Соответственно для решения оптимизационной задачи требуется знание первой производной $f(x)$ в аналитической форме.

Шаг перехода λ в сторону минимума:

$$\lambda = \text{const} \quad \text{(может расходиться)}$$
$$\lambda_k = \lambda_{k-1}/p \quad \text{дробится (уменьшается) на каждом шаге}$$
$$\lambda_k = \underset{\lambda_k}{\text{argmin}}\, f\left(\mathbf{x}_k + \lambda_k \nabla f\left(\mathbf{x}_k\right)\right)$$

Последний случай - метод наискорейшего спуска, который обеспечивает наискорейшее схождение алгоритма к минимуму функции

Выход по значению функции

Критерий выхода по значению функци означает, что мы приблизились к вычислению некоторого локального минимума с заранее заданной точностью, т.е.:

$$\left| f\left(\mathbf{x}_k\right) - f\left(\mathbf{x}_{k-1}\right) \right| < \varepsilon_{\mathrm{f}}$$

Выход по значению аргумента

Критерий выхода по аргументу означает, что мы приблизились к значению аргумента $\mathbf{x}$, соответствующего некоторому локальному минимуму с заранее заданной точностью, т.е.:

$$\left\| \mathbf{x}_k - \mathbf{x}_{k-1} \right\| < \varepsilon_{\mathrm{x}}$$

Выход по объему вычислений

Итерационный процесс останавливается при достижении некоторого количества итераций и/или количества вычислений функции $f(\mathbf{x}_k)$.

Метод покоординатного спуска (метод Гаусса-Зейделя)

Метод n-мерной оптимизации, в котором поиск минимума функции осуществляется последовательно по каждой компоненте вектора $\mathbf{x}$, что в итерационным виде записывается как:

$$x_{i,\,k+1} = x_{i,\,k} + \lambda_{i,\,k} \nabla f_i \left(\mathbf{x}_k \right)$$

Алгоритм метода:

1. Фиксируем x_1 и находим минимум только для этой компоненты.
2. Делаем все аналогично для каждой компоненты $\mathbf{x}$ для $i = 1, 2, \ldots, n$.
3. Повторяем пп. 1-2 до тех пор, пока критерии выхода не выполнятся.
   (здесь осуществляется переход к $k + 1$)

λ_i вычисляется для каждого x_i

Метод наискорейшего спуска

Метод n-мерной оптимизации, в котором поиск минимума осуществляется одновремен
по всем компонентам, что в итерационным виде записывается как:

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \lambda_k \nabla f\left(\mathbf{x}_k\right)$$

а λ_k находится решением своей оптимизационной задачи:

$$\lambda_k = \underset{\lambda_k}{\operatorname{argmin}}\, f\left(\mathbf{x}_k + \lambda_k \nabla f\left(\mathbf{x}_k\right)\right)$$

Если λ определяется НЕ решением оптимизационной задачи, то это простой (градиентный) метод первого порядка

Методы второго порядка (методы Ньютона)

Методы n-мерной оптимизации, в которых величина шага поиска минимума определяется величиной второй производной функции $f(x)$, что в итерационным виде записывается как:

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \mathbf{H}^{-1}\left(\mathbf{x}_k\right) \nabla f\left(\mathbf{x}_k\right),$$

где $\mathbf{H}$ - матрица Гессе, определяемая как:

$$\mathbf{H}\left(\mathbf{x}_k\right) = \begin{bmatrix} \frac{\partial^2 f(x)}{\partial x_1^2} & \cdots & \frac{\partial^2 f(x)}{\partial x_1 \partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial^2 f(x)}{\partial x_n \partial x_1} & \cdots & \frac{\partial^2 f(x)}{\partial x_n \partial x_n} \end{bmatrix}$$

Требуется знание вторых производных $f(\mathbf{x})$ в аналитическом виде.
$\mathbf{H}$ сложно вычислять, поэтому переходят к квазиньютоновским методам

Метод Ньютона-Рафсона

Метод n-мерной оптимизации, в котором шаг перехода выбирается оптимальным (по аналогии с методом наискорейшего спуска - методом первого порядка), итерационный вид записывается как:

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \lambda_k \mathbf{H}^{-1}\left(\mathbf{x}_k\right) \nabla f\left(\mathbf{x}_k\right),$$

а λ_k находится решением своей оптимизационной задачи:

$$\lambda_k = \underset{\lambda_k}{\operatorname{argmin}} f\left(\mathbf{x}_k + \lambda_k \mathbf{H}^{-1}\left(\mathbf{x}_k\right) \nabla f\left(\mathbf{x}_k\right)\right)$$

В целях снижения объема вычислений $\mathbf{H}$ можно вычислять не на каждом шаге итерационного процесса, а один раз на m шагов

На практике могут использовать более сложные алгоритмические методы, которые, как правило, имеют более быструю сходимости к экстремуму:

- методы первого порядка:
  - метод сопряженных градиентов;
    (дополнительно учитываем направление движения решения)
    (метод первого порядка для нелинейного функционала)
- методы второго порядка:
  - квазиньютоновские методы;
    (упрощение вычисления Гессиана)
  - метод Левенберга-Марквардта
    (один из основных методов решения задач МНК)
- другие методы:
- метод Нелдера-Мида;
- генетические алгоритмы;
- методы Монте-Карло.

Метод наименьших квадратов

Один из наиболее часто используемых методов математического формирования задачи оптимизации, в котором целевая функция J представляет собой интегральную меру квадрата отклонения (ошибки) e некоторой функции $f(\mathbf{x})$ от заданной y.

$$\sum_i e^2 = \sum_i \Big(f(\mathbf{x}_i) - y_i \Big)^2 \longrightarrow \min$$

Задачи, зачастую, решаемые МНК:

- аппроксимация экспериментальных данных;
- идентификация моделей динамических систем.

Задача аппроксимации экспериментальных данных

Задача аппроксимации экспериментальных данных заключается в численном подборе параметров (составляют вектор θ) функции $f(\theta, t)$, наиболее точно описывающих (повторяющих) характер изменения набора данных $y(t)$, полученного экспериментально. Критерий минимизации формируется МНК, как достижение минимума квадрата рассогласования функции с подбираемыми параметрами $f(\theta, t)$ относительно полученных значений экспериментально $y(t)$, т.е.:

$$\theta = \underset{\theta \in \Theta}{\operatorname{argmin}} \sum_i \Big(f(\theta, t_i) - y_i(t_i) \Big)^2$$

`fminsearch` - поиск минимума функции;
`fmincon` - поиск минимума функции с ограничениями;
`fit` - аппроксимация набора данных простыми функциями;
`lsqcurvefit` - аппроксимация набора данных НМНК.